1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE 20.Jan.99	3. REPORT TYPE AND DATES COVERED DISSERTATION

4. TITLE AND SUBTITLE
A 3-D UNSTRUCTURED CFD METHOD FOR MANEUVERING VECHICLES.

5. FUNDING NUMBERS

6. AUTHOR(S)
MAJ HUGHSON MONTGOMERY C

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
MISSISSIPPI STATE UNIVERSITY

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
THE DEPARTMENT OF THE AIR FORCE
AFIT/CIA, BLDG 125
2950 P STREET
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

FY99-53

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited distribution
In Accordance With AFI 35-205/AFIT Sup 1

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

DTIC QUALITY INSPECTED 4

19990219124

14. SUBJECT TERMS

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

A 3-D UNSTRUCTURED CFD METHOD

FOR MANEUVERING VEHICLES

By

Montgomery C. Hughson

A Dissertation
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Aerospace Engineering
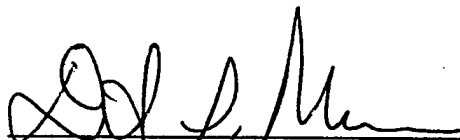in the Department of Aerospace Engineering
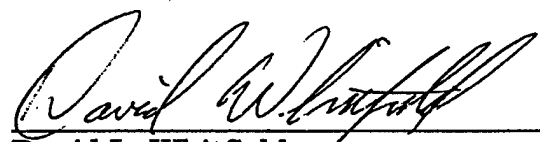
Mississippi State, Mississippi

December 1998

A 3–D UNSTRUCTURED CFD METHOD

FOR MANEUVERING VEHICLES

By

Montgomery Hughson
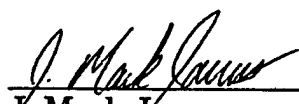
Approved:
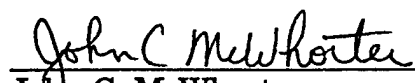
_____
David L. Marcum
Professor of Mechanical Engineering
(Director of Dissertation)

_____
David L. Whitfield
Professor of Aerospace Engineering
(Major Professor)
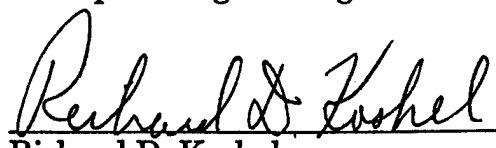
_____
W. Roger Briley
Professor of Mechanical Engineering
(Committee Member)

_____
Bharat K. Soni
Professor of Aerospace Engineering
(Committee Member)

_____
J. Mark Janus
Associate Professor of
Aerospace Engineering
(Committee Member)

_____
John C. McWhorter
Professor, Graduate Coordinator, and
Head of the Department of
Aerospace Engineering

_____
A. Wayne Bennett
Dean of the College of Engineering

_____
Richard D. Koshel
Dean of the Graduate School

Name: Montgomery C. Hughson

Date of Degree: December 18, 1998

Institution: Mississippi State University

Major Field: Engineering (Aerospace Engineering)

Director of Dissertation: Dr. David L. Marcum

Major Professor: Dr. David L. Whitfield

Title of Study: A 3–D UNSTRUCTURED CFD METHOD
FOR MANEUVERING VEHICLES

Pages in Study: 113

Candidate for Degree of Doctor of Philosophy

Numerical simulation of maneuvering vehicles is accomplished using a three–dimensional (3–D) unstructured computational fluid dynamic (CFD) method. The equations of fluid motion used are either the inviscid Euler equations or the full viscous Navier–Stokes equations cast in an Arbitrary Lagrangian–Eulerian (ALE) framework. A turbulence model developed by Spalart and Allmaras is used for viscous solutions. The system of fluid equations are solved implicitly using upwind, flux–splitting techniques for the convective fluxes of either Roe or Van Leer with up to second–order temporal and spatial accuracy for steady or unsteady computations. Innovative boundary conditions for a moving mesh to include inviscid, viscous, far–field and a solid rocket motor exhaust exit surface were developed. The temporal solution is found using an application of Newton's method. The computational field simulation (CFS) of two 3–D wings and a waisted–body of revolution are compared to experimental data for boundary condition validation. An unsteady CFS of a pitching wing is validated by comparison to experimental data. A number of unsteady missile maneuver trajectories coupled with a six degree–of–freedom model using Euler angles and the Flat–Earth model are presented.

DEDICATION

To my wife, Mary Ann, and my mother, Felicitas Sylva (Langlet).

takes he made. But of course, you learn more from your mistakes than your successes. Finally, I could not have completed this endeavor without "standing on the shoulders" of an MSU alumni, Dr. W. Kyle Anderson, who allowed me to start code development from his existing steady 3–D unstructured flow solver. He is an amazing and talented researcher who gave freely of his time and encouragement.

A task of this magnitude could not have been completed without the outstanding support I received from the staff at the Engineering Research Center. The administrative people who keep the computers running and up-to-date with the latest operating systems and software are truly the unsung heroes. Let me sing their praises.

Thank you all for your support.

Montgomery C. "Monty" Hughson
Mississippi State University
December, 1998.

LIST OF FIGURES

vi

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

The first recorded use of computational fluid dynamics (CFD) can be traced back more than 90 years ago, almost two generations, to 1917 when the British scientist L. F. Richardson attempted weather prediction by solving the governing partial differential equations numerically – by hand [1]. Since then, phenomenal advances have been made with the rapid development of the computer and related numerical algorithms which has made these type of solutions easier and faster to obtain. Throughout, the overarching objective has been to develop CFD methods for predicting the outcome of an important dynamic event – like the weather. Another critically important problem is the prediction of a maneuvering vehicle's trajectory. The development of a 3–D unstructured CFD method for trajectory prediction through numerical simulation of the flowfield about maneuvering vehicles is documented herein. The impetus for such techniques are as follows.

Consider the development of an advanced fighter aircraft such as the United States Air Force (USAF) F–22A. In July 1997 the flyaway cost per aircraft was estimated to be $92.6 million and its IOC (Initial Operational Capability) was projected to be in November 2004 – a mere 23 years after the USAF identified the requirement. Contrast this with the flyaway cost of the F–86A *Sabre* which was $0.2 million and it was 39 months between the first flight of the XP–86 and the *Sabre's* combat debut in Korea [2]. In a little over a few decades RDT&E (Research, Development, Test and Evaluation) costs and

1

schedule lengths have grown by 2 to 3 orders of magnitude. Much of the cost and schedule increase is directly attributable to the complexity of the technology being incorporated as well as the high cost of building and flight testing a high performance fighter aircraft. Well before the real aircraft is built extensive wind–tunnel testing is done on physical models in an iterative way to establish a design to meet aerodynamic requirements. These costs are also quite high and the model building and testing process is time–consuming. For a flight vehicle development program as basic as a tactical missile design, once the aerodynamic data is available it takes at least six months and 1–4 man–years of effort to develop a trajectory simulation [3]. These are areas in the RDT&E process where CFD methods can be used to conduct virtual simulations at greatly reduced time and cost. This is the primary motivation for the development of a 3–D unstructured CFD method for maneuvering vehicles.

The following sections discuss the current issues in dynamic motion CFD, a brief history of unstructured mesh flow solver development, a review of current dynamic motion unstructured CFD methods, and concludes with an outline of the following chapters.

1.1 Dynamic Motion CFD Issues

The numerical simulation, or virtual simulation, of a maneuvering vehicle refers to a dynamic motion CFD problem in aerodynamics. These type problems are "defined as the simulation of flowfields exhibiting unsteadiness on the physical scale of the vehicle. Examples include, but are not limited to, store and missile stage separation, unsteady airfoil [and wing] motion, aeroelasticity, and inlet unstart [4]." The virtual simulations could consist of, but are not limited to, determining the flowfield about the vehicle of interest, predicting the effect of the flowfield on the vehicle – induced motion and/or de-

formation through aeroelastic effects, moving the vehicle and/or changing its shape, and then going back to the beginning to determine the flowfield about the vehicle in its new position.

The work in this study addresses two major issues for 3–D dynamic motion unstructured CFD predictions – a flow solver for a moving mesh which accurately captures the physics of the fluid motion about the vehicle and a rigid body model which allows for dynamic motion on the order of the vehicle scale. Actually, in the unstructured CFD method developed herein, the two issues are intertwined. The flow solver allows for vehicle–scale translations and rotations through the use of boundary conditions whereby the numerical mesh, rigidly attached to the vehicle, is allowed to maneuver freely wherever the vehicle's trajectory may lead. This is different from other unstructured CFD efforts, as far as this author is aware, which peg the boundary of the entire simulation computational domain to a fixed position. Using internal mesh deformations the vehicle's motion is simulated through a prescribed maneuver or by quasi–steady or unsteady aerodynamic forces and moments. This necessitates schemes to manage the mesh deformations and mesh speeds in a way that drives these values to zero at the stationary boundary. It also requires computational domains, that is, numerical meshes, large enough to warrant those boundary conditions. One can easily see how this confines or restricts the time and distance a simulation can run on a particular mesh and requires more node points than may actually be required to capture the physics and the trajectory.

On the other hand, a rigidly attached mesh which moves with the body requires constant updating of all the geometric terms, or metrics, at each time–step whereas the deforming mesh metric updates are usually restricted

to the region nearest the body – which is deforming the most. However, for a full Navier–Stokes calculation to capture the essential physics, the boundary layer mesh should be kept rigid and orthogonal to the vehicle body surface to avoid any mesh deformation effects. The mesh in the region closest to the body, where the node points can be thought of as the "observers" of the complex physical phenomena near the surface, must be kept in rigid alignment to report the physics accurately. Eventually, to handle relative motion and multiple bodies, a global mesh would be required with all the inherent mesh management requirements mentioned. The integration of the 3–D unstructured CFD method developed herein for each body should, theoretically and conceptually, present no major problems.

Another equally important issue is the time–step restrictions imposed by the fluid dynamics. The time–scales for the fluid dynamics are usually orders of magnitude smaller than those for the rigid body dynamics. Techniques to allow larger time–stepping, such as implicit flow solvers and boundary conditions, are employed herein. In many cases there exists a small region of the numerical field that restricts the time–steps for time–accurate predictions. This may be a place where the numerical mesh cells are quite small in order to resolve the physics, such as a boundary layer, or a region whose imposed boundary condition involves large gradients with subsequent small time–steps. A technique to allow local time–stepping in these areas thereby increasing the global minimum time–step is also employed which arguably does not affect the global time accuracy.

1.2 Unstructured Mesh Technology

The use of unstructured meshes in the finite–volume CFD community was driven by the need for a Computational Field Simulation (CFS) over com-

plex geometries with complex physics. It is well known unstructured mesh generation provides greater flexibility and ease of adaptation to flow features than structured mesh generation. Unstructured mesh generation technology is also far more amenable to automation than the multiblock strategies employed in structured meshes for complex geometries. On the other hand, structured mesh flow solvers have a long history of development compared to the unstructured mesh flow solvers and have achieved remarkable efficiencies and speed–ups. The unstructured mesh flow solvers require more memory to handle additional overhead managing the mesh data structure which are not necessary with the implicit connectivity of a structured mesh. In the final analysis, as stated by Venkatakrishnan [5], "with the best of the unstructured grid flow solvers, the computational costs are between two and three times those associated with a structured grid with the same number of unknowns." This is highly encouraging, in the author's opinion, because as the complexity of the geometry goes up the number of nodes (unknowns) required for a structured mesh grows to many times more than that required for an unstructured mesh. For excellent discussion and reviews of unstructured mesh generation and flow solver technology see the AGARD Report [6], the VKI lecture series by Mavriplis [7], and the survey article by Venkatakrishnan [5]. A brief history of unstructured mesh flow solver development from reference [5] follows.

Almost 13 years ago, in January 1986, Jameson *et al.* [8] presented the first full 3–D solution about a complete Boeing 747–200 for inviscid transonic flow using an unstructured mesh and flow solver. At around the same time, the structured flow solver community was actively pursuing upwind schemes by using the Van Leer [9] monotonic upstream–centered scheme for conservation laws (MUSCL) and the approximate Riemann solvers of Roe [10]. These

were eventually implemented on the unstructured side by numerous researchers such as Anderson [11], Barth and Jesperson [12], Batina [13], and Frink [14], to name just a few.

Turbulence modeling is required for realistic applications at high Reynolds numbers. Marcum and Agarwal [15] implemented two versions of $k-\epsilon$ models for turbulent flow over an ONERA M6 wing. Recently, the one–equation models of Baldwin–Barth [16] and Spalart–Allmaras [17] have become the turbulence models of choice for unstructured mesh flow solvers. Analysis and comparison of these two can be found in references [18] and [19].

1.3 Unstructured CFD and Dynamic Motion

In the summer of 1996 the Air Force Office of Scientific Research (AFOSR) sponsored the First AFOSR Conference on Dynamic Motion CFD. This meeting served as a review of the work to date in unstructured and structured CFD methods for virtual simulations of unsteady flows of aerodynamic interest [4]. There were many impressive unstructured and structured methods and results presented. A review of a few of the unstructured results follows. For details concerning the structured mesh technology results see the proceedings in reference [4].

Lohner *et al.* [20] showed explicit time–stepping Euler simulations of a hypersonic store release at Mach = 8 (free motion) and a fuel tank release from an F–16 (quasi–steady) as well as weapon fragmentation with a large number of objects. They also presented a rotating missile solution comparison between a rigid and a flexible body assumption. The simulation technique involved a deforming, stationary mesh. Baysal *et al.* [21] employed their Dynamic Unstructured Technique (DUT), which uses explicit four–stage Runge–Kutta time–stepping, to simulate store separation (quasi–steady) with

a mesh rigidly attached to the store moving through a stationary global mesh. An adaptation window was placed about the store and the interface between it and the global mesh was deformed and remeshed. Venkatakrishnan and Mavriplis [22] presented subsonic inviscid simulation of a multi–element airfoil system during deployment and transonic turbulent flow simulations of a self–excited shock oscillation on a 18% thick circular–arc airfoil and a 2–element slotted airfoil with flap rotation. They used an agglomeration multigrid procedure for implicit time–accuracy with a multi–stage Runge–Kutta scheme as an explicit smoother within the multigrid algorithm. They also used a spring analogy for deformation of the stationary boundary global mesh and found it insufficient for viscous solutions. Their solution was to use a distance function in the "viscous" region to maintain a region about the airfoil which moves rigidly attached to the airfoil surface.

At NASA Langley Research Center, Kleb [23] has used an implicit time–integration scheme, which uses a Gauss–Seidel procedure, and an inviscid algorithm to simulate a subsonic pitch–over programmed maneuver for a Vertical Take–off and Vertical Landing (VTVL) flight vehicle. He employed a spring analogy for mesh movement within a stationary mesh. His results-showed significant differences between an unsteady and a quasi–steady trajectory simulation.

As far as the author is aware, the 3–D unstructured CFD method for maneuvering vehicles developed in this study is a first application of a number of ideas and techniques to these type problems. An implicit Newton's method is used for unsteady time–accurate simulation of a transonic, compressible flow trajectory driven by aerodynamic and thrust–vectoring forces and moments. A rigidly attached mesh is allowed to move freely with the ve-

hicle body over large distances without the restriction of a stationary mesh and inherent mesh deformations and management overhead involved therein. As will be shown in the results, a thrust–vectoring schedule can be prescribed to allow truly stand–alone vehicle maneuvers.

1.4 Outline

The work documented in the following chapters is organized as follows: Chapter II presents the equations of fluid motion, their frame of reference, and the turbulence model. Chapter III details the derivation of the Flat–Earth equations and the Six–Degree–of–Freedom (6DOF) model using the equations of rigid body motion and dynamic and kinematic analysis. The spatial and temporal discretization as well as the solution scheme for the system of fluid equations are discussed in Chapters IV and V, respectively. Chapter VI presents the boundary conditions for moving mesh considerations. Chapter VII consists of the 3–D unstructured CFD method for dynamic motion validation results and discussion. A summary and conclusions are in Chapter VIII. Finally, an Appendix is included to document the inviscid flux Jacobians employed in the numerical scheme which, like a large dataset, were just too big to include in the main text but are of vital importance to anyone wishing to duplicate this work.

CHAPTER II
EQUATIONS OF FLUID MOTION

In this study the fluid flow is assumed to be governed by the time–dependent Reynolds–averaged Navier–Stokes equations cast in an arbitrary Lagrangian–Eulerian (ALE) frame of reference to account for the dynamic motion of the mesh. In the sections to follow, the governing equations will be presented in their entirety, followed by brief discussions of the ALE method and the Spalart–Allmaras turbulence model.

2.1 Governing Equations

The governing equations of fluid motion are given as a system of conservation laws which relate the time rate of change of the conserved quantities – mass, momentum, and energy – in a control volume to the flux of these quantities through the boundary of the control volume. The equations are nondimensionalized by the free stream density, ρ_∞, speed of sound, a_∞, temperature, T_∞, viscosity, μ_∞, thermal conductivity, k_∞, and a reference length, L. For a bounded domain, or control volume, Ω with boundary $\partial\Omega$ the governing equations can be written in integral form as

$$\frac{\partial}{\partial t}\int_\Omega Q\, d\mathcal{V} + \oint_{\partial\Omega} F(Q)\cdot\hat{n}\, dS = \oint_{\partial\Omega} F^v(Q)\cdot\hat{n}\, dS \qquad (2.1)$$

where $\hat{n}$ is the outward–pointing unit normal for the control volume and the vector Q of the conserved variables of mass, momentum, and energy is

$$Q = \left[\, \rho,\ \rho u,\ \rho v,\ \rho w,\ E \,\right]^T \qquad (2.2)$$

9

For a constant control volume (for example, a rigid, non–deforming mesh) the governing equations of Equation (2.1) can be written

$$\mathcal{V}\frac{\partial Q}{\partial t} + \oint_{\partial\Omega} F(Q)\cdot\hat{n}\,dS = \oint_{\partial\Omega} F^{v}(Q)\cdot\hat{n}\,dS \tag{2.3}$$

For a moving mesh with no time variation of the volume the geometric conservation law (GCL) is satisfied. The GCL is of the same integral form as the mass conservation law and a numerical scheme satisfies the GCL if a spatially uniform flow is preserved on an arbitrary moving mesh [24]. In the validation sections to follow it will be shown that arbitrary movement for a rigid mesh in uniform flow does not induce errors in the computed flow.

The flux vectors are

$$F = f\hat{i} + g\hat{j} + h\hat{k} \quad \text{and} \quad F^{v} = \frac{M_{\infty}}{\text{Re}_{L}}\left(f^{v}\hat{i} + g^{v}\hat{j} + h^{v}\hat{k}\right) \tag{2.4}$$

where $\hat{i}, \hat{j}, \hat{k}$ are the unit vectors in the x, y, z directions, respectively, M_{∞} is the freestream Mach number and Re_{L} is the Reynolds number, a measure of the ratio of inertia forces to viscous forces, which is defined as

$$\text{Re}_{L} = \frac{\rho_{\infty}|\vec{V}_{\infty}|L}{\mu_{\infty}} \tag{2.5}$$

where μ_{∞} is the freestream coefficient of viscosity. The inviscid (f, g, h) and viscous (f^{v}, g^{v}, h^{v}) fluxes are defined as follows

$$f = \begin{bmatrix} \rho U \\ \rho U u + p \\ \rho U v \\ \rho U w \\ (E + p)U + x_{t}p \end{bmatrix}, \quad g = \begin{bmatrix} \rho V \\ \rho V u \\ \rho V v + p \\ \rho V w \\ (E + p)V + y_{t}p \end{bmatrix}, \quad h = \begin{bmatrix} \rho W \\ \rho W u \\ \rho W v \\ \rho W w + p \\ (E + p)W + z_{t}p \end{bmatrix} \tag{2.6}$$

$$f^{v} = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - \dot{q}_{x} \end{bmatrix}, \quad g^{v} = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{yz} \\ u\tau_{xy} + v\tau_{yy} + w\tau_{yz} - \dot{q}_{y} \end{bmatrix} \tag{2.7}$$

$$h^{v} = \begin{bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ u\tau_{xz} + v\tau_{yz} + w\tau_{zz} - \dot{q}_{z} \end{bmatrix}$$

In the inviscid fluxes the contravariant velocities, U, V, and W, are defined by

$$U = u - x_t, \quad V = v - y_t, \quad W = w - z_t \tag{2.8}$$

where x_t, y_t, and z_t are the mesh speeds in the x, y, and z directions, respectively. Hence the velocity vector of a fluid particle, $\vec{V}$, is written relative to the motion of the mesh with the contravariant velocities of Equation (2.8) as the velocity components in the x, y, z directions, respectively. The contravariant face speed is defined as

$$a_t = x_t n_x + y_t n_y + z_t n_z \tag{2.9}$$

where n_x, n_y, and n_z are the components of the outward–pointing unit vector, $\hat{n}$, in the x, y, z directions, respectively. The set of equations for the inviscid fluxes is closed by an equation of state for a perfect gas. Hence, the pressure p is given by

$$p = (\gamma - 1)\left[E - \frac{1}{2}\rho(u^2 + v^2 + w^2)\right] \tag{2.10}$$

where γ is the ratio of specific heats and is assigned a value of 1.4 for air.

For the viscous fluxes the stress components are defined as follows:

$$\tau_{xx} = (\mu + \mu_T)\left[2u_x - \frac{2}{3}(u_x + v_y + w_z)\right]$$
$$\tau_{yy} = (\mu + \mu_T)\left[2v_y - \frac{2}{3}(u_x + v_y + w_z)\right] \tag{2.11a}$$
$$\tau_{zz} = (\mu + \mu_T)\left[2w_z - \frac{2}{3}(u_x + v_y + w_z)\right]$$

and

$$\tau_{xy} = \tau_{yx} = (\mu + \mu_T)(u_y + v_x)$$
$$\tau_{xz} = \tau_{zx} = (\mu + \mu_T)(u_z + w_x) \tag{2.11b}$$
$$\tau_{yz} = \tau_{zy} = (\mu + \mu_T)(v_z + w_y)$$

and the heat flux is

$$\nabla\dot{q} = -\frac{1}{\gamma - 1}\left(\frac{\mu}{\text{Pr}} + \frac{\mu_T}{\text{Pr}_T}\right)\nabla T \tag{2.12}$$

where μ and μ_T are the laminar and turbulent viscosities and Pr and Pr$_T$ are the laminar and turbulent Prandtl numbers. The laminar viscosity is computed using Sutherland's Law which is

$$\frac{\mu}{\mu_\infty} = \frac{(1 + C^\cdot)}{(T/T_\infty + C^\cdot)}(T/T_\infty)^{3/2} \qquad (2.13)$$

where $C^\cdot = 110.33/T_\infty$ is Sutherland's constant (the numerator constant as given by Warsi in [25]) divided by a freestream reference temperature which is assumed to be in $^\cdot K$. The turbulent viscosity is computed by solving the Spalart–Allmaras one–equation turbulence model [17] which will be discussed in a later section. The laminar and turbulent Prandtl numbers are given values of 0.72 and 0.9, respectively [26].

2.2 Arbitrary Lagrangian–Eulerian (ALE) Method

The ALE method was first described in the 1974 paper of Hirt *et al.* [27] and has been used by many researchers since then to handle unstructured CFD problems with moving meshes. Publication of a number of results within the past few years using the Arbitrary–Lagrangian Eulerian technique can be found in [20] – [23] and [28] – [37].

The ALE method is a combination of Lagrangian and Eulerian methods which essentially allows the mesh motion to be chosen arbitrarily and is perfect for simulations where the mesh motion is not predetermined. In the Lagrangian frame of reference the computational mesh is attached to the fluid particles and moves with the local fluid velocity while in the Eulerian frame the mesh is fixed in space and the fluid particles move from one control volume, or cell, to another. Hence, in the Lagrangian method each cell is associated with the same fluid element whereas in the Eulerian method the fluid flows through each cell. The inviscid (f, g, h) fluxes of Equation (2.6) are re-

peated below to show how one recovers either frame from the ALE formulation.

$$f = \begin{bmatrix} \rho U \\ \rho U u + p \\ \rho U v \\ \rho U w \\ (E+p)U + x_t p \end{bmatrix}, \quad g = \begin{bmatrix} \rho V \\ \rho V u \\ \rho V v + p \\ \rho V w \\ (E+p)V + y_t p \end{bmatrix}, \quad h = \begin{bmatrix} \rho W \\ \rho W u \\ \rho W v \\ \rho W w + p \\ (E+p)W + z_t p \end{bmatrix}$$

Recall that the mesh speeds were contained in the contravariant velocity terms, defined in Equation (2.8) repeated below

$$U = u - x_t, \quad V = v - y_t, \quad W = w - z_t$$

If the mesh is fixed then the mesh speeds are zero and the Eulerian form of the inviscid fluxes are recovered, that is

$$f = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u w \\ (E+p)u \end{bmatrix}, \quad g = \begin{bmatrix} \rho v \\ \rho v u \\ \rho v^2 + p \\ \rho v w \\ (E+p)v \end{bmatrix}, \quad h = \begin{bmatrix} \rho w \\ \rho w u \\ \rho w v \\ \rho w^2 + p \\ (E+p)w \end{bmatrix} \tag{2.14}$$

If the mesh speed equals the fluid velocity then the contravariant velocities are zero and the Lagrangian form of the inviscid fluxes are recovered whereby the advective terms vanish identically [38].

2.3 Spalart–Allmaras Turbulence Model

In this study, the one–equation turbulence model of Spalart–Allmaras is used [17]. The form of this equation is given by

$$\begin{aligned}
\frac{D(\tilde{v})}{Dt} &= c_{b1}[1 - f_{t2}]\tilde{S}\tilde{v} + \frac{M_\infty}{\sigma \mathrm{Re}}\left\{ \nabla \cdot \left[(v + (1 + c_{b2})\tilde{v})\nabla\tilde{v} - c_{b2}\tilde{v}\nabla^2\tilde{v} \right] \right\} \\
&\quad - \frac{M_\infty}{\mathrm{Re}}\left(c_w f_w - \frac{c_{b1}}{\kappa^2}f_{t2} \right)\left(\frac{\tilde{v}}{d} \right)^2 + \frac{\mathrm{Re}}{M_\infty}f_{t1}\Delta U^2
\end{aligned} \tag{2.15}$$

where

$$f_{v1} \equiv \frac{\chi^3}{\chi^3 + c_{v1}^3} \quad \chi \equiv \frac{\tilde{v}}{v} \quad \tilde{S} \equiv S f_{v3} + \frac{M_\infty}{\mathrm{Re}}\frac{\tilde{v}}{\kappa^2 d^2}f_{v2} \quad f_{v2} \equiv 1 - \frac{\chi}{1 + \chi f_{v1}}$$

and

$$f_{v3} \equiv \frac{(1 + \chi f_{v1})(1 - f_{v2})}{\chi} \quad f_w \equiv g\left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}} \right)^{1/6} \quad g \equiv r + c_{w2}(r^6 - r) \quad r \equiv \frac{M_\infty}{\mathrm{Re}}\frac{\tilde{v}}{S\kappa^2 d^2}$$

In these equations, d is the distance to the nearest wall and S is the magnitude of the vorticity, that is

$$S = \sqrt{(w_y - v_z)^2 + (u_z - w_x)^2 + (v_x - u_y)^2} \qquad (2.16)$$

For a moving mesh the left–hand side of Equation (2.15), the material derivative of the eddy viscosity, $\nu_t = \tilde{\nu}f_{\nu 1}$, is

$$\frac{D\nu_t}{Dt} \equiv \frac{\partial \nu_t}{\partial t} + (U_i - x_{t_i})\frac{\partial \nu_t}{\partial x_i} \qquad (2.17)$$

where U_i and x_{t_i} are the fluid velocity and mesh speed in the respective Cartesian directions. Furthermore, the definitions of f_{t2} and f_{t1} are

$$f_{t2} = c_{t3}\exp(-c_{t4}\chi^2) \qquad f_{t1} = c_{t1}g_t\exp(-c_{t2}\frac{\omega_t^2}{\Delta U^2}[d^2 + g_t^2 d_t^2]) \qquad (2.18)$$

The last term in Equation (2.15) is used when specifying a transition location where f_{t1} is the trip function. In this trip function, Spalart and Allmaras [17] state that "d_t is the distance from the field point to the trip, and ΔU is the difference between the velocity at the field point and that at the trip. Then $g_t \equiv \min(0.1, \Delta U/\omega_t\Delta x)$ where Δx is the grid spacing along the wall at the trip." The implicit turbulent flow solver used in this study, obtained from Anderson and Bonhaus [39] at NASA Langley Research Center, includes this term but all the computations were done assuming fully turbulent flow.

Finally, for completeness, the constants are given as $c_{b1} = 0.1355$, $\sigma = 2/3$, $c_{b2} = 0.622$, $\kappa = 0.41$, $c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma$, $c_{w2} = 0.3$, $c_{w3} = 2.0$, $c_{\nu 1} = 7.1$, $c_{t1} = 1$, $c_{t2} = 2$, $c_{t3} = 1.2$, $c_{t4} = 0.5$.

For a complete explanation of the precise meaning and the detailed derivation of each term in the Spalart–Allmaras one–equation turbulence model for aerodynamic flows as well as comparisons to existing models such as Baldwin–Barth and Johnson–King for a number of different cases, the reader is referred to [17].

After Equation (2.15) is solved for $\tilde{\nu}$, the eddy viscosity is evaluated as

$$\mu_t = \rho\nu_t = \rho\tilde{\nu}f_{\nu 1} \qquad (2.19)$$

The value of $\tilde{\nu}$ is specified to be zero on the body. In the far–field, the dependent variable is the free stream value for inflow and is extrapolated from the interior for outflow. To solve the one–equation model, discretization of the convective terms uses first–order upwinding and the higher–order derivatives are evaluated as described in Chapter IV. The temporal discretization is described in Chapter V.

CHAPTER III

EQUATIONS OF RIGID BODY MOTION

A detailed derivation of the equations of rigid body motion will be presented in this chapter which constitute the six degree–of–freedom (6DOF) model used in this study. The next section serves as an overview of the 6DOF model. The following sections present dynamic and kinematic analyses used to derive the state equations. The last section is a summary of the set of equations derived in this chapter, the Flat–Earth equations, which are the basis for the 6DOF model.

3.1 Six Degree–of–Freedom (6DOF) Model

There are many good engineering mechanics references for the kinematics and kinetics of rigid bodies which list and describe the equations of rigid body motion [40], [41]. For the majority of the equations to be presented, particularly as they apply to maneuvering flight vehicles, the text by Stevens *et al.* [42] was used extensively.

The notion of a rigid body implies the vehicle's shape remains constant with no deformation such as bending, twisting or stretching. Hence, for a numerical mesh that constitutes a mathematical, or computational, model of the body of interest all the mesh points maintain fixed relative positions in space at all times – although their inertial positions may be changing. The type of rigid body considered in this study will be a flight vehicle – a manned or unmanned aircraft or a missile. The equations of motion for this vehicle can be

16

decoupled into rotational and translational equations if the coordinate origin is situated at the vehicle's center of mass. The center of mass is more commonly referred to as the center of gravity (*CG*) which is the term used herein. The rotational motion of the flight vehicle, say an aircraft, is then the same as pitching (nose up), yawing (nose right), and rolling (right wing down) motion about the *CG* as if it were fixed in space. The translational motion consists of the three components of the translation of the *CG*. These constitute the six degrees of freedom (6DOF) for the maneuvering vehicle.

A state vector can be designed which contains 12 variables which completely define the vehicle's motion. Three components of position determine the vehicle's potential energy in a gravitational field. Three components of velocity specify the translational kinetic energy and three components of angular velocity specify the rotational kinetic energy. Lastly, three attitude variables are needed to specify the vehicle's orientation relative to the gravity vector in the inertial frame. Therefore, these 12 components constitute the vehicle's state at any instant during a simulation. A set of four equations — force, moment, attitude and navigation — will be derived which allow prediction of the vehicle's state at the next instant from current state information.

Before deriving the necessary equations it is important to get a visual picture of a flight vehicle's frames of reference and orientations. Consider the aircraft shown in Figure 3.1. An axes system aligned backwards, up, and left relative to the aircraft will be used for development of the kinematic relations. This system is traditionally used for wind tunnel data because drag, *D*, lift, *L*, and cross-wind force, *C*, are defined naturally in these axes as the aerodynamic force components along the positive *x*, *y*, and *z* axes, respectively.
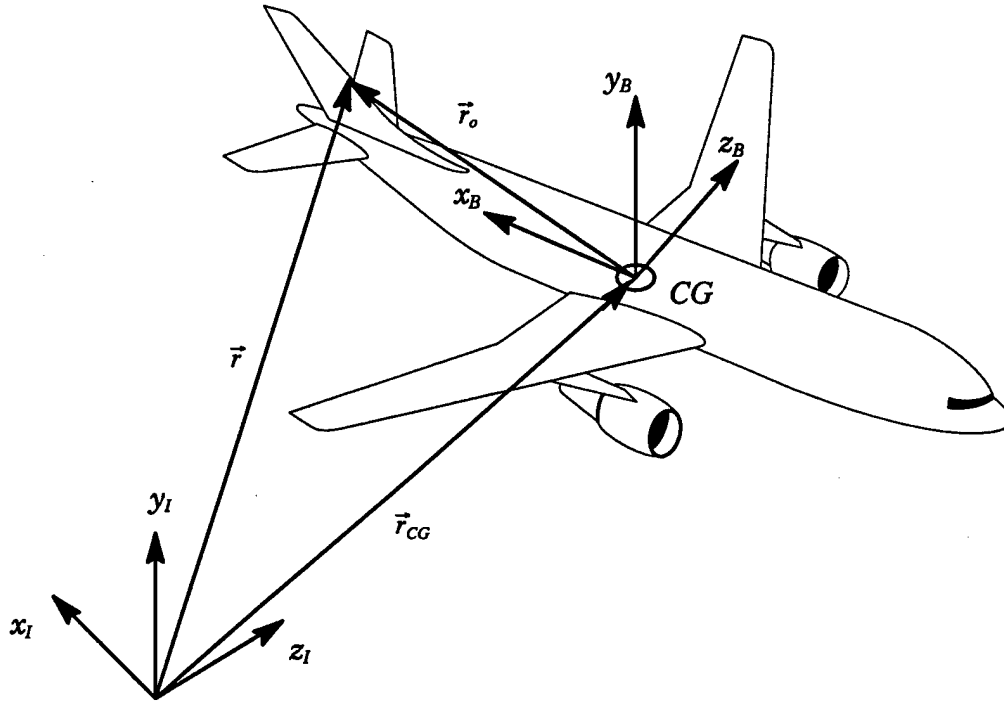
Figure 3.1 Definition of flight vehicle axes

Figure 3.1 shows the two sets of flight vehicle axes used in this study for development of the 6DOF model. One set of axes will be fixed in space which is the inertial frame and variables referenced in this frame will be denoted by subscript I. The other set of axes moves freely through inertial space and is rigidly attached to the vehicle body frame with its origin at the vehicle CG. Variables referenced in this frame will be denoted by subscript B. The position in inertial space of any point on the vehicle body is defined by the relative position vector, denoted with a subscript o, from the CG to that point. For a rigid body the length of the relative position vector remains constant. Thus, the position vector of any point on the flight vehicle in the inertial frame can be written as

$$\vec{r} = \vec{r}_{CG} + \vec{r}_o \qquad (3.1)$$

The velocity of any point on the flight vehicle can then be determined by taking the derivative of the position vector, which is

$$\dot{\vec{r}} = \vec{v}_{CG} + \vec{\omega} \times \vec{r}_o \tag{3.2}$$

where the angular velocity term is

$$\vec{\omega} = \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \tag{3.3}$$

and P, Q, and R are the components of the angular velocity of the vehicle body frame in inertial coordinates in the x, y, and z directions, respectively.

In the following two sections, dynamic and kinematic analysis will be used to derive the four state equations.

3.2 Dynamic Analysis

The translational and rotational equations of motion (EOM) for the vehicle can be determined by applying Newton's second law. The derivation of these EOM's will be presented in detail in the following paragraphs.

Newton's second law, applied to translational motion, relates force to rate of change of linear momentum. In this study there are three forces involved – aerodynamic, propulsion, and gravity. The aerodynamic forces depend on the vehicle shape and the airflow about it while the propulsion forces are reaction forces on the vehicle caused by the propulsion system. Hence, it is better to apply Newton's second law in the reference frame attached to the vehicle body with origin at the CG. Then the translational EOM is given by

$$\vec{F}_B + Bm\vec{g} = \frac{d}{dt_I}(m\vec{v}_B) \tag{3.4}$$

where $\vec{F}_B$ is the vector sum of the aerodynamic and propulsion forces with the subscript B indicating these are in the vehicle body frame, $Bm\vec{g}$ is the gravity force rotated into the body frame by the rotation matrix B (presented in the

next section), and $\frac{d}{dt_I}(m\vec{v}_B)$ is the time rate of change of the linear momentum of the vehicle taken with respect to the inertial reference frame. Expanding this last term yields

$$\frac{d}{dt_I}(m\vec{v}_B) = \dot{m}\vec{v}_B + m\frac{d}{dt_I}(\vec{v}_B) \tag{3.5}$$

For some flight vehicles, missiles for example, the $\dot{m}$ term can be important, but for aircraft it is negligible. This term will be dropped in this analysis so that the translational EOM is simply

$$\vec{F}_B + Bm\vec{g} = m\frac{d}{dt_I}(\vec{v}_B) \tag{3.6}$$

The time derivative can be expanded as

$$\frac{d}{dt_I}(\vec{v}_B) = \dot{\vec{v}}_B + \vec{\omega}_B \times \vec{v}_B \tag{3.7}$$

If the equation is divided through by the vehicle mass and the acceleration term is moved to the left–hand side then the force equation, the first of the set of four state equations, is obtained and given by

$$\dot{\vec{v}}_B = -\Omega_B\vec{v}_B + B\vec{g} + \frac{\vec{F}_B}{m} \tag{3.8}$$

where

$$\Omega_B = \begin{bmatrix} 0 & -R & Q \\ R & 0 & -P \\ -Q & P & 0 \end{bmatrix} \tag{3.9}$$

is the cross–product matrix, which is defined by

$$\vec{\omega}_B \times \vec{v}_B \equiv \Omega_B\vec{v}_B \tag{3.10}$$

and, referring to Figure (3.1), the gravity vector in the inertial frame is

$$\vec{g} = \begin{bmatrix} 0 \\ -g_o \\ 0 \end{bmatrix} \tag{3.11}$$

Newton's second law, applied to rotational motion, relates the resultant external moment about the CG, or the torque force, to the time rate of change of the angular momentum. The angular EOM is given by

$$\vec{T}_B = \frac{d}{dt_I}(\vec{H}_B) \tag{3.12}$$

where $\vec{T}_B$ is the net torque acting about the vehicle body CG, $\vec{H}_B$ is the angular momentum vector of the rigid vehicle, and the time derivative is taken with respect to the inertial frame. For a flight vehicle the torque is generated by the aerodynamic control surfaces, reaction control thrusters, or by any part of the engine thrust not acting through the CG – thrust vectoring being a good example. The torque vector is defined as

$$\vec{T}_B = \begin{bmatrix} L \\ M \\ N \end{bmatrix} \tag{3.13}$$

where L is used for the rolling moment component to distinguish it from lift, L. In this study, the components of this vector come from the aerodynamic moment calculations. An expansion of the time derivative term yields

$$\frac{d}{dt_I}(\vec{H}_B) = \dot{\vec{H}}_B + \vec{\omega}_B \times \vec{H}_B \tag{3.14}$$

The angular momentum is given by

$$\vec{H}_B = J_B\vec{\omega}_B \tag{3.15}$$

where J_B is the inertia matrix of the rigid body which is

$$J = \begin{bmatrix} J_{xx} & -J_{xy} & -J_{xz} \\ -J_{xy} & J_{yy} & -J_{yz} \\ -J_{xz} & -J_{yz} & J_{zz} \end{bmatrix} \tag{3.16}$$

Terms of the inertia matrix are computed, for example, as

$$moment\ of\ inertia\ about\ x-axis = J_{xx} = \int (y^2 + z^2)dm$$
$$cross-product\ of\ inertia\ \ J_{xy} \equiv J_{yx} = \int xy\,dm \tag{3.17}$$

A set of axes, called the principal axes, can be selected such that the cross-products of inertia become zero and the inertia matrix becomes diagonal. Most flight vehicles are symmetric about the x–y plane so that for every xz or yz in the cross–product of inertia integral there is a product identical in magnitude but opposite in sign. Hence, only the J_{xy} (and, of course, J_{yx}) cross–product of

inertia is nonzero and the inertia matrix has four zero entries. The time deriv-
ative of the angular momentum is

$$\dot{\vec{H}}_B = \dot{J}_B\vec{\omega}_B + J_B\dot{\vec{\omega}} \tag{3.18}$$

When finding derivatives of the angular momentum with respect to time it is
easier to fix the reference frame to the vehicle to avoid varying moments and
cross–products of inertia which then makes the inertia matrix constant. After
some rearrangement the second of the 4 state equations, the moment equa-
tion, is obtained

$$\dot{\vec{\omega}}_B = -J_B^{-1}\Omega_B J_B\vec{\omega}_B + J_B^{-1}\vec{T}_B \tag{3.19}$$

The fourth equation of the four state equations, the navigation equa-
tion, is easily obtained from kinematics as

$$\dot{\vec{P}}_I = B^T\vec{V}_B \tag{3.20}$$

The left–hand side of the navigation equation is the time derivative in the in-
ertial frame of the vehicle position, denoted by $\vec{P}$, or the vehicle's velocity in
the inertial frame, which is

$$\dot{\vec{P}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{3.21}$$

and B^T is the transpose, which is the same as the inverse, of the rotation ma-
trix.

The next section will develop the third of the four state equations, the
attitude equation as well as B, the rotation matrix which takes a vector from
the inertial frame to the vehicle body frame.

3.3 Kinematic Analysis

In this section the coordinate rotation matrix from the inertial frame to
the vehicle body frame will be developed. A fundamental kinematic relation-
ship for relative rotation between two coordinate frames, called the strapdown

equation in guidance and navigation computations, will then be used to develop the third of the four state equations, the attitude equation.

Referring back to Figure 3.1, recall the vehicle body frame is aligned x–backward (from the nose of the aircraft), y–up, and z–port (the left side of an aircraft or missile looking forward). This is known as the wind tunnel axes system due to its common use in the reduction and presentation of wind tunnel data as discussed previously. In the aircraft industry there is an accepted sequence of rotations to describe the instantaneous attitude of the flight vehicle with respect to the inertial frame at any instant during a trajectory simulation. Starting from the inertial frame the sequence is as follows [42]:

1. Rotate about the y–axis, nose right (positive yaw ψ)

2. Rotate about the new z–axis, nose up (positive pitch θ)

3. Rotate about the new x–axis, right wing down (positive roll ϕ)

If going from the body frame back to the inertial frame, the sequence is done in reverse order – roll, pitch, and yaw. The yaw, pitch and roll angles ψ, θ, ϕ are known as the Euler angles. In terms of coordinate rotations, or transformations, the relationship between the vehicle body frame and the inertial frame can be written as

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_B = B_\phi B_\theta B_\psi \begin{bmatrix} x \\ y \\ z \end{bmatrix}_I \tag{3.22}$$

Figure 3.2 shows a rotation of an aircraft's nose to the right to graphically illustrate the construction of the individual rotation matrix for yaw which would be

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_B = \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_I = B_\psi \begin{bmatrix} x \\ y \\ z \end{bmatrix}_I \tag{3.23}$$

Particular features of the plane rotation matrix of Equation (3.23) to note are that it approaches the identity matrix as ψ approaches zero, contains sine and cosine terms, and leaves the y–component unchanged. The other individual rotation matrices for pitch and roll can be determined by inspection in a similar manner as B_ψ. Equation (3.22) can then be written as

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_I \tag{3.24}$$
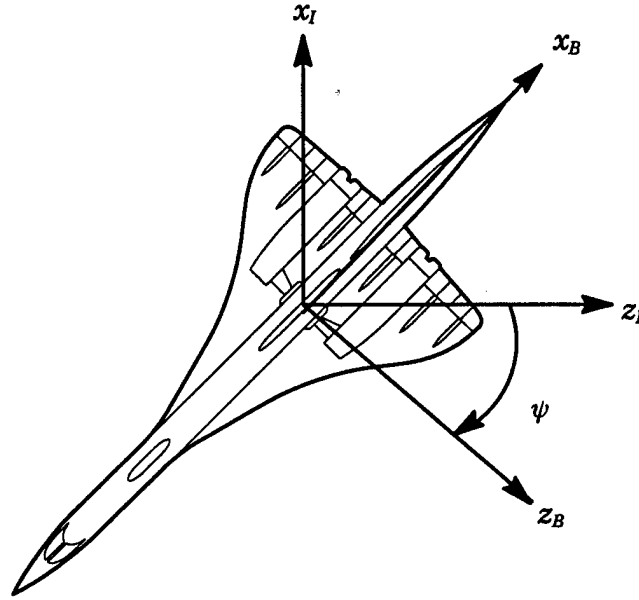
Figure 3.2 Rotate nose right from inertial to body frame

The rotation matrix B provides for the complete transformation of a vector from the inertial frame to the vehicle body frame. It is the matrix multiplication result of the individual rotations which can now be written as

$$B = B_\phi B_\theta B_\psi = \begin{bmatrix} \cos\theta\cos\psi & -\sin\theta & \cos\theta\sin\psi \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\cos\theta & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\cos\theta & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi \end{bmatrix} \tag{3.25}$$

To transform from the vehicle body frame to the inertial frame involves using the inverse of the rotation matrix. Since the rotation matrix is orthogonal its inverse is simply the transpose, that is

$$B^{-1} = B^T \qquad (3.26)$$

The rotation matrix satisfies the strapdown equation (see reference [42] for a derivation of this relationship) which is given as

$$\dot{B} = -\Omega_B B \qquad (3.27)$$

where Ω_B is the cross–product matrix given in Equation (3.9). The strapdown equation yields nine coupled differential equations. From these a set of three state equations expressing the Euler angle rates as functions of the Euler angles and the body–axes angular rates can be derived. Using Equations (3.25) and (3.27) and evaluating elements (1,2), (1,3), and (3,2) yields the following equations for the Euler angle derivatives

$$\frac{d}{dt}\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} -1 & -\tan\theta\cos\phi & -\tan\theta\sin\phi \\ 0 & \sin\phi & -\cos\phi \\ 0 & -\cos\phi/\cos\theta & -\sin\phi/\cos\theta \end{bmatrix}\begin{bmatrix} P \\ Q \\ R \end{bmatrix}_B \qquad (3.28)$$

or, in more compact notation,

$$\dot{\vec{\Phi}} = \mathcal{B}(\Phi)\vec{\omega}_B, \quad \Phi^T = [\phi,\ \theta,\ \psi] \qquad (3.29)$$

The set of equations represented in Equations (3.28) and (3.29) are the third set of the four state equations, the attitude equations.

There are a few disadvantages to using the three–variable Euler angles for attitude orientation which is readily apparent by a close examination of Equation (3.28). First of all, if the pitch angle θ reaches $\pm$ 90° there is a division by zero. In digital simulations, reaching $\pm$ 90° exactly rarely happens, but numerical difficulties also arise in the vicinity of those value. Secondly, the Euler angles themselves may integrate up to angles outside the standard $\pm$ 90° range of pitch and the standard $\pm$ 180° range of the bank and yaw angles.

This leads to uniqueness problems when determining the vehicle's attitude. Finally, the equations are linear in P, Q, and R, but nonlinear in terms of the Euler angles. These problems can be overcome by using a four–variable attitude orientation. These four variables are known as quaternions [42] and they would be required for simulations of around–the–Earth flight, all–attitude flight and simulations of spinning bodies.

For the trajectory simulations considered in this study the Euler angles will be sufficient. The final set of equations using the Euler angles are commonly referred to as the Flat–Earth equations [42]. These equations and the state vector will be summarized in the next section for easy reference. A short discussion of some features of this equation set will also be provided.

3.4 Flat–Earth Equations

The so–called Flat–Earth equations have been derived in the previous sections and are summarized as follows

$$\dot{\vec{v}}_B = -\Omega_B \vec{v}_B + B\vec{g} + \frac{\vec{F}_B}{m} \qquad \text{(force)} \qquad (3.30a)$$

$$\dot{\vec{\omega}}_B = -J_B^{-1}\Omega_B J_B \vec{\omega}_B + J_B^{-1}\vec{T}_B \qquad \text{(moment)} \qquad (3.30b)$$

$$\dot{\vec{\Phi}} = \mathcal{S}(\Phi)\vec{\omega}_B \qquad \text{(attitude)} \qquad (3.30c)$$

$$\dot{\vec{P}}_I = B^T\vec{v}_B \qquad \text{(navigation)} \qquad (3.30d)$$

where the vehicle's state at a particular instant is now completely defined by the state vector, which is

$$\vec{X}^T = \left[\vec{v}_B^T, \vec{\omega}_B^T, \vec{\Phi}^T, \vec{P}_I\right] \qquad (3.31)$$

As pointed out by Stevens *et al.* in [42] there are nonlinearities and coupling in these equations which should be mentioned and discussed.

The force equation requires $\vec{\omega}_B$ and $\vec{\Phi}$ from the other equations for the coefficients in the Ω_B and B matrices and is driven by the external gravity and

body forces. The moment equation is nonlinear in the first term, couples into the other equations through $\vec{\omega}_B$, and is driven externally by the applied torque. The attitude equation is driven by $\vec{\omega}_B$ from the moment equation and is nonlinear. The navigation equation is uncoupled from the other equations. Finally, note that terms in the equations are non–dimensionalized using the same dimensional variables as those used for the governing fluid equations.

Aerodynamic force and moment calculations provide the body force and torque terms necessary to drive these equations for a trajectory simulation. These force and moment results are the integrated result of pressure and skin friction forces acting on the vehicle surface and are referenced to the vehicle's *CG*. In the axes system used in this study, the force calculations are in the inertial frame and the moment calculations are in the vehicle body frame. These are the natural frames of reference for the wind–tunnel axes system. Appropriate rotations to either frame are accomplished using the rotation matrix. Many texts, such as references [43] and [44], discuss the calculation of the forces and moments from the pressure and skin friction distributions. These should be referred to for details.

CHAPTER IV

SPATIAL DISCRETIZATION

The spatial discretization of the system of fluid equations will be presented. In the following sections the finite–volume approach will be used to evaluate the inviscid and viscous flux contributions, upwind schemes for the inviscid fluxes will be discussed, and methods for schemes higher than first–order spatial accuracy as well as limiters will be discussed.

4.1 Finite–Volume Scheme

The system of equations of fluid motion in Equation (2.3), which is repeated below, are discretized using a node–centered, edge–based, finite–volume approach.

$$\mathcal{V}\frac{\partial Q}{\partial t} + \oint_{\partial \Omega} F(Q)\cdot\hat{n}dS = \oint_{\partial \Omega} F^{v}(Q)\cdot\hat{n}dS$$

In this study, the domain of interest is divided into a finite number of tetrahedra, although, from a strictly edge–based point of view any 3–D polygonal shape can be used. Control volumes are constructed about each node, or vertex, of the numerical mesh by connecting the centroid of each tetrahedron, or cell, to the centroid of each of the four triangular faces of the cell and to the midpoint of the six edges of the cell. The net result is division of the cell into four equal sub–volumes. In 2–D the control volumes are constructed by connecting the centroid of the triangular face to the midpoint of the three edges dividing the face into three equal areas. The resulting mesh of nodes at centroids and midpoints with non–overlapping volumes is called the dual mesh.

28
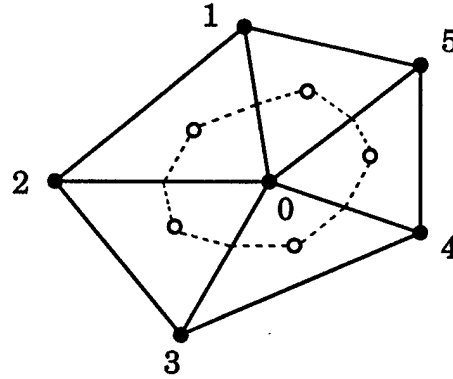
Figure 4.1 Control volume surrounding node

Figure 4.1 is a representation of the control volume surrounding a node for a 2–D case. The numerical evaluation of the surface integrals contained in Equation (2.3) is handled separately for the inviscid and the viscous contributions.

In a finite–volume formulation, the inviscid flux contribution can be approximated using midpoint evaluation of the flux over each surface (edge in 2–D, refer to Figure 4.1) that constitutes the bounding surfaces of the dual mesh control volume. Hence, the inviscid flux evaluation of Equation (2.3) can be written as

$$\oint_{\partial\Omega} F(Q)\cdot\hat{n}_d dS_d = \oint_{\partial\Omega} F(Q)\cdot\vec{dn}_d \approx \sum_{i=1}^{N_d} \Phi\left(Q^+, Q^-; \hat{n}_{d_i}\right) a_{d_i} \tag{4.1}$$

The N_d limit represents the number of dual mesh surfaces of the control volume about the node of interest and a_{d_i} is the area of that surface. The numerical flux function, $\Phi(Q^+, Q^-; \hat{n}_{d_i})$, is constructed from values on the left (Q^+) and right (Q^-) sides of the surface. These values are extrapolated from surrounding data and details on this procedure will be discussed in a later section. The

left and right sides are determined by the outward–pointing normal, $\hat{n}$, which points from the left side of the surface (inside) to the right side (outside). The so–called edge–based (edge–surface based) formulation exploits these facts by looping over each edge, determining the flux contribution at each edge–surface, and distributing these to adjacent nodes thereby accumulating, or summing, the net contribution at each surface about all the nodes. Hence, the contribution to each surface is added to the control volume 'inside' and subtracted from the control volume on the 'outside'.

An improvement in computational efficiency can be obtained by replacing the directed areas from the dual mesh that join at the midpoint of an edge in the tetrahedral mesh with a single directed area [45]. The resulting average directed area is the same as the area vector normal to the line formed by connecting the adjoining cell centroids. For a numerical flux of first–order spatial accuracy on the face a simple average of the fluxes at the two adjacent nodes can be used,

$$\Phi(Q^+, Q^-; \hat{n}) = \tfrac{1}{2}(F(Q_L) + F(Q_R)) \cdot \hat{n} \tag{4.2}$$

where Q^+ is the value at the left node, Q_L, and Q^- is the value at the right node, Q_R. Techniques to obtain higher order estimates of Q^+ and Q^- will be discussed in a later section. Upwind schemes for the inviscid flux evaluations will be discussed in the following section.

A finite–volume formulation for the viscous flux surface integral evaluations involve terms such as those in Equations (2.7) and (2.11), which are of the form

$$\oint_{\partial\Omega} [(\mu + \mu_T)(u_y + v_x)] \cdot \hat{n}_x dS \tag{4.3}$$

The laminar and turbulent viscosity term, $(\mu + \mu_T)$, is evaluated as an average of the surrounding nodes. Gradient terms at the nodes themselves are calculated using Green's Theorem. The evaluation of the viscous terms with this approach yields identical formulas as a Galerkin approximation as given by Barth in reference [46].

4.2 Upwind Schemes

In modern CFD, upwind schemes have been developed to "numerically simulate more properly the direction of propagation of information in a flow field along characteristic curves" [47]. This development was driven by the problems central–difference schemes experienced with flow discontinuities such as shocks, slip–lines and contact surfaces. In this study, two upwind flux splitting techniques are used – Roe's Flux Difference Splitting (FDS) [10] and van Leer's Flux Vector Splitting (FVS) [48]. The extension of van Leer's FVS technique for unsteady computations on a moving mesh can be found in references [37], and [49] to [52]. The derivation of these splittings can be found in the references. The following two sections will present the split flux results as used herein.

4.2.1 Roe's Flux Difference Splitting (FDS)

A numerical flux on the edge–surface of a control volume constructed using Roe's approximate Riemann solver is formed as

$$\Phi = \frac{1}{2}\left(F(q^+;\hat{n}) + F(q^-;\hat{n})\right) - \frac{1}{2}|A(\tilde{q};\hat{n})|(q^+ - q^-) \tag{4.4}$$

This numerical flux can also be written as

$$\Phi = \frac{1}{2}\left(F(q_L) + F(q_R)\right)\cdot\hat{n} - \frac{1}{2}|A(\tilde{q};\hat{n})|(q^+ - q^-) \tag{4.5}$$

In Equation (4.5), $F(q_L)$ and $F(q_R)$ are the flux vectors evaluated from values at the nodes on each side of the edge instead of from data extrapolated to the edge. Furthermore, in Equations (4.4) and (4.5), note that

$$q = \begin{bmatrix} \rho \\ u \\ v \\ w \\ p \end{bmatrix} \tag{4.6}$$

is the vector of primitive variables and $\tilde{q}$ in the Roe–matrix, A, is the primitive variables constructed from the Roe–averaged variables which are

$$\tilde{u} = \frac{\sqrt{\rho^+}\,u^+ + \sqrt{\rho^-}\,u^-}{\sqrt{\rho^+} + \sqrt{\rho^-}}$$

$$\tilde{v} = \frac{\sqrt{\rho^+}\,v^+ + \sqrt{\rho^-}\,v^-}{\sqrt{\rho^+} + \sqrt{\rho^-}}$$

$$\tilde{w} = \frac{\sqrt{\rho^+}\,w^+ + \sqrt{\rho^-}\,w^-}{\sqrt{\rho^+} + \sqrt{\rho^-}} \tag{4.7}$$

$$\tilde{H} = \frac{\sqrt{\rho^+}\,H^+ + \sqrt{\rho^-}\,H^-}{\sqrt{\rho^+} + \sqrt{\rho^-}}$$

$$\tilde{a} = (\gamma - 1)\left[\tilde{H} - \frac{1}{2}\left(\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2\right)\right]$$

and H is the total enthalpy which is

$$H = \frac{E + p}{\rho} \tag{4.8}$$

These as well as the flux Jacobian Roe–matrix A can also be found in many texts such as references [53] and [54]. Roe's scheme for a moving mesh problem is particularly easy to implement from a non–moving mesh formulation by redefining the Roe–averaged contravariant velocity as [37]

$$\tilde{U} = \tilde{u}n_x + \tilde{v}n_y + \tilde{w}n_z - a_t \tag{4.9}$$

where a_t is the contravariant face speed given in Equation (2.9).

The solution of the linearized Riemann problem involves discontinuous jumps and entropy violating expansion, or rarefaction, waves result when

these waves are transonic or sonic [1], [53], [54]. An entropy fix proposed by Harten and Hyman [55] to avoid this expansion shock "consists in introducing a local expansion fan in the approximate Riemann solution when an expansion is detected through a sonic point" [54]. The modulus of the eigenvalue $|\lambda|$ of the flux Jacobian matrix for that wave is corrected by

$$|\lambda| = \begin{cases} |\lambda| & \text{if } |\lambda| \geqq \varepsilon \\ \frac{1}{2}\left(\frac{\lambda^2 + \varepsilon^2}{\varepsilon}\right) & \text{if } |\lambda| < \varepsilon \end{cases} \qquad (4.10)$$

where

$$\varepsilon = \max\left[0, (\lambda - \lambda^+), (\lambda^- - \lambda)\right] \qquad (4.11)$$

4.2.2 Van Leer's Flux Vector Splitting (FVS)

For van Leer's FVS, the flux vectors are formed in terms of the Mach number normal to the face – the contravariant Mach number – which is

$$M_n = \frac{U}{a_\infty} \qquad (4.12)$$

where the contravariant velocity is

$$U = \vec{V} \cdot \hat{n} = un_x + vn_y + wn_z - a_t \qquad (4.13)$$

The fluxes are then evaluated according to whether the flow is supersonic or subsonic through the control volume face. For supersonic flow in the direction of a face normal, $M_n \geqq 1$, the fluxes are evaluated as

$$\hat{F}^+ = \left[F(q) \cdot \hat{n}\right]^+ = F, \qquad \hat{F}^- = \left[F(q) \cdot \hat{n}\right]^- = 0 \qquad (4.14)$$

and for supersonic flow in the opposite direction of a face normal, $M_n \leqq -1$, as

$$\hat{F}^+ = \left[F(q) \cdot \hat{n}\right]^+ = 0, \qquad \hat{F}^- = \left[F(q) \cdot \hat{n}\right]^- = F \qquad (4.15)$$

In a subsonic flow case, $|M_n| < 1$, the fluxes are split into two contributions $\hat{F}^+$ and $\hat{F}^-$. This splitting is done in such a way that the Jacobian matrix of $\hat{F}^+$ has positive eigenvalues and the Jacobian matrix of $\hat{F}^-$ has negative eigenvalues. The split fluxes are evaluated as

$$F = \hat{F}^+(q^-) + \hat{F}^-(q^+) \qquad (4.16)$$

The split fluxes are given by

$$\hat{F}^{\pm} = \begin{cases} f_{mass}^{\pm} \\ f_{mass}^{\pm}[u + n_x(-\overline{U} \pm 2a)/\gamma] \\ f_{mass}^{\pm}[v + n_y(-\overline{U} \pm 2a)/\gamma] \\ f_{mass}^{\pm}[w + n_z(-\overline{U} \pm 2a)/\gamma] \\ f_{energy}^{\pm} \end{cases} \tag{4.17}$$

with

$$f_{mass}^{\pm} = \pm \frac{\rho a}{4}(M_n \pm 1)^2 \tag{4.18}$$

and

$$f_{energy}^{\pm} = f_{mass}^{\pm}\left[\frac{(1 - \gamma)\overline{U}^2 \pm 2(\gamma - 1)\overline{U}a + 2a^2}{(\gamma^2 - 1)} + \frac{u^2 + v^2 + w^2}{2} + \frac{a_t(-\overline{U} \pm 2a)}{\gamma}\right] \tag{4.19}$$

4.3 Higher–Order Methods

To evaluate the flux vector it is necessary to extrapolate flow field values to the control volume face or edge from surrounding node values. First–order spatial accuracy is obtained if the flow field value at the node on that side of the face is used to evaluate the flux at the edge. To obtain higher–order spatial accuracy piecewise linear variable reconstruction is commonly used in unstructured flow solvers for the extrapolation [45]. The primitive variables, $q^T = [\rho, u, v, w, p]$, are extrapolated rather than the conserved variables, $Q^T = [\rho, \rho u, \rho v, \rho w, E]$, to keep negative pressures from occurring.

Piecewise linear reconstruction is obtained by using a Taylor's series expansion for a function of multiple variables which is

$$q(x_i, y_i, z_i) = q(x_o, y_o, z_o) + \nabla q(x_o, y_o, z_o) \cdot \Delta\vec{r} + O(\Delta\vec{r}^2) \tag{4.20}$$

where ∇q is the gradient of q and $\Delta\vec{r}$ is the vector from the node at (x_o, y_o, z_o) to the desired point at (x_i, y_i, z_i). The two methods used in this study to obtain the gradient, ∇q, are based on least–squares and Green's theorem discussed in the next two sections. The final section discusses the limiter used in this study.
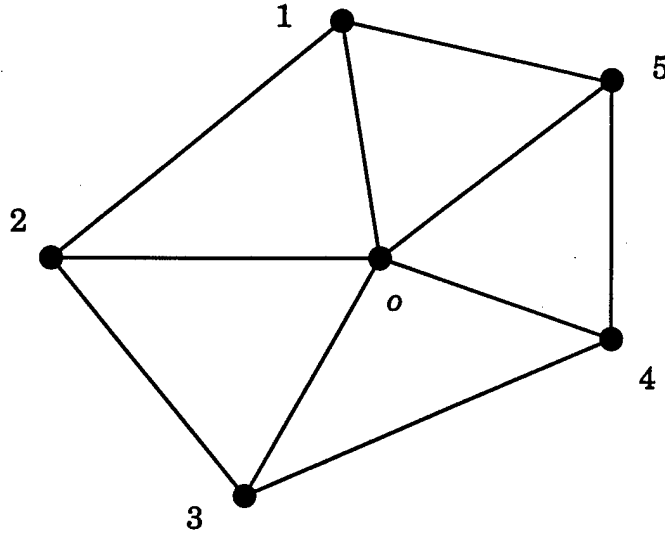
Figure 4.2 Least–squares reconstruction data

4.3.1 Least–Squares

The least–squares approach uses data from surrounding nodes and the assumption the data behaves linearly in this region to estimate a gradient at the node o. Figure 4.2 shows a node o with 5 surrounding, or adjacent, nodes. All the node data will be used to construct a gradient at node o. The data at each node i can be expressed as

$$q_i = q_o + q_{x_o}(x_i - x_o) + q_{y_o}(y_i - y_o) + q_{z_o}(z_i - z_o), \quad i = 1,...,5 \qquad (4.21)$$

For the nodes shown in Fig. 4.2, a 5 x 3 system of equations is formed which can be solved for the gradient components at node o which can be written as

$$\begin{bmatrix} \Delta x_1 & \Delta y_1 & \Delta z_1 \\ \Delta x_2 & \Delta y_2 & \Delta z_2 \\ \Delta x_3 & \Delta y_3 & \Delta z_3 \\ \Delta x_4 & \Delta y_4 & \Delta z_4 \\ \Delta x_5 & \Delta y_5 & \Delta z_5 \end{bmatrix} \begin{bmatrix} q_{x_o} \\ q_{y_o} \\ q_{z_o} \end{bmatrix} = \begin{bmatrix} \Delta q_1 \\ \Delta q_2 \\ \Delta q_3 \\ \Delta q_4 \\ \Delta q_5 \end{bmatrix} \qquad (4.22)$$

where, for example,

$$\Delta x_1 = x_1 - x_o, \quad \Delta q_1 = q_1 - q_o \qquad (4.23)$$

This over–determined system of linear equations, $Ax = b$, can be solved using the normal equation approach by multiplying both sides by the transpose of the coefficient matrix, A, to obtain a 3 x 3 system of equations.

$$A^T A x = A^T b \qquad (4.24)$$

As pointed out by Anderson and Bonhaus [45], for grids which are highly stretched the accuracy of the normal equation approach to the least–squares solution is degraded because the sensitivity of the solution is dependent on the square of the condition number of A. In other words, the numerical solution using the normal equation approach is susceptible to roundoff error [57]. To circumvent this problem, Anderson and Bonhaus [45] employ a Gram–Schmidt QR decomposition process which is used in this study as well. Additional methods for solving linear least–squares problems are Singular Value Decomposition (SVD) and Householder transformations [57]. These were not implemented in this study.

In the Gram–Schmidt QR process the coefficient matrix A is decomposed into a product of an orthogonal matrix Q, and an upper triangular matrix R, that is

$$Ax = QRx = b \qquad (4.25)$$

and the solution of the system becomes

$$x = R^{-1} Q^T b \qquad (4.26)$$

The Gram–Schmidt process was selected over the other methods because it allows for the easy precomputation and storage of weight factors so gradients at each node can be evaluated by looping over the edges in the mesh and distributing the contribution of each edge to each of the nodes [45]. The formulas for calculating the gradients at node o are

$$q_{x_o} = \sum_{i=1}^{N} W_i^x \Delta q_i \qquad q_{y_o} = \sum_{i=1}^{N} W_i^y \Delta q_i \qquad q_{z_o} = \sum_{i=1}^{N} W_i^z \Delta q_i \qquad (4.27)$$

The summation is over all the nodes adjacent to (the edges that connect to) node o ($N = 5$ in Figure 4.2). The weight factors are given by

$$W_i^x = \frac{\Delta x_i}{r_{11}^2} - \frac{r_{12}}{r_{11}r_{22}^2}\left(\Delta y_i - \Delta x_i\frac{r_{12}}{r_{11}}\right) + \frac{(r_{12}r_{23} - r_{13}r_{22})}{r_{11}r_{22}r_{33}}\left[\Delta z_i - \Delta x_i\frac{r_{13}}{r_{11}} - \frac{r_{23}}{r_{22}}\left(\Delta y_i - \Delta x_i\frac{r_{12}}{r_{11}}\right)\right] \quad (4.28)$$

$$W_i^y = \frac{1}{r_{22}^2}\left(\Delta y_i - \Delta x_i\frac{r_{12}}{r_{11}}\right) - \frac{r_{23}}{r_{22}r_{33}^2}\left[\Delta z_i - \Delta x_i\frac{r_{13}}{r_{11}} - \frac{r_{23}}{r_{22}}\left(\Delta y_i - \Delta x_i\frac{r_{12}}{r_{11}}\right)\right] \quad (4.29)$$

$$W_i^z = \frac{1}{r_{33}^2}\left[\Delta z_i - \Delta x_i\frac{r_{13}}{r_{11}} - \frac{r_{23}}{r_{22}}\left(\Delta y_i - \Delta x_i\frac{r_{12}}{r_{11}}\right)\right] \quad (4.30)$$

where

$$r_{11} = \left(\sum_{i=1}^{N}\Delta x_i^2\right)^{\frac{1}{2}} \qquad r_{12} = \frac{\sum_{i=1}^{N}\Delta x_i\Delta y_i}{r_{11}} \qquad r_{13} = \frac{\sum_{i=1}^{N}\Delta x_i\Delta z_i}{r_{11}} \quad (4.31)$$

$$r_{22} = \left[\sum_{i=1}^{N}\left(\Delta y_i - \Delta x_i\frac{r_{12}}{r_{11}}\right)^2\right]^{\frac{1}{2}} \qquad r_{23} = \frac{\sum_{i=1}^{N}\Delta z_i\left(\Delta y_i - \Delta x_i\frac{r_{12}}{r_{11}}\right)}{r_{22}} \quad (4.32)$$

$$r_{33} = \left\{\sum_{i=1}^{N}\left[\Delta z_i - \Delta x_i\frac{r_{13}}{r_{11}} - \frac{r_{23}}{r_{22}}\left(\Delta y_i - \Delta x_i\frac{r_{12}}{r_{11}}\right)\right]^2\right\}^{\frac{1}{2}} \quad (4.33)$$

Equation (4.22) is an unweighted least squares system in which all the data surrounding node o are given equal consideration. However, the elements of the coefficient matrix A are components of the distance vector to node i and consequently the least squares process tends to de–emphasize more distant data when compared to the Green's theorem method [58].

4.3.2 Green's Theorem

The Green's theorem approach can also be used to estimate a gradient at the node o. Green's theorem is

$$\int_{\Omega} \nabla q\, d\mathcal{V} = \oint_{\partial\Omega} q\hat{n}\, dS \quad (4.34)$$

Referring back to Figure 4.1, the control volume about node o is the dual mesh volume. With the assumption the gradient is constant throughout the dual

volume, Green's formula for the gradient can be evaluated by numerically integrating along the dual mesh boundary and dividing the result by the dual volume. The right–hand side of Equation (4.34) can be approximated using the trapezoidal rule, which is exact for piecewise linear q [59], and written as

$$\oint_{\partial\Omega} q\hat{n}dS = \sum_{j=1}^{N_S} \frac{1}{2}(q_j + q_{j+1})\hat{n}_j \tag{4.35}$$

where N_S is the total number of endpoints of the sides of the dual mesh about node o, q_j and q_{j+1} are the values of q at each endpoint of a side, and $\hat{n}_j$ is the area vector perpendicular to the side. Barth [60] has shown how to transform and rearrange the summation to put it into a form particularly amenable to the edge–based data structure. The final formula for the gradient at node o can then be written as

$$\nabla q_o = \frac{1}{\mathcal{V}} \sum_{i=1}^{N} \frac{1}{2}(q_i + q_o)\hat{n}_i \tag{4.36}$$

where $\mathcal{V}$ is the dual volume, the summation is over all the nodes adjacent to (the edges that connect to) node o ($N = 5$ in Figure 4.2), and the $\hat{n}_i$ in the sum are the individual area vectors normal to the lines formed by connecting the cell centers to the midpoints of each edge of the adjoining cells.

4.3.3 Barth–Jesperson Limiter

The Barth–Jesperson limiter [12] is used to control oscillations in the numerical solution which may occur near steep gradients. A limited form of Equation (4.20) is implemented as

$$q_{face} = q_o + \Psi_{face}\nabla q_o \cdot \Delta\vec{r} \tag{4.37}$$

where Ψ is a variable that varies from zero (first–order spatial accuracy) to one thereby limiting the flux.

The value for Ψ is determined as

$$
\Psi_i = \begin{cases} \min\left(1, \dfrac{q_o^{\max} - q_o}{q_i - q_o}\right), & \text{if } q_i - q_o > 0 \\[2ex] \min\left(1, \dfrac{q_o^{\min} - q_o}{q_i - q_o}\right), & \text{if } q_i - q_o < 0 \\[2ex] 1 & \text{if } q_i - q_o < 0 \end{cases} \tag{4.38}
$$

where $q_o^{\max}$ and $q_o^{\min}$ are the maximum and minimum value of q among the adjacent nodes and node o, q_i is the adjacent node on the other side of the face of interest, and hence Ψ_i is the limiter value for the face in that direction. In this study, each q_i is limited separately as opposed to using a single value for Ψ for all. This limiter invokes the monotonicity principle in that values of the linearly reconstructed data do not exceed the maximum and minimum of neighboring values [12].

An additional limiter due to Venkatakrishnan [61] is available in the code. During the course of development of the flow solver, it appeared this limiter produced varying pressure distributions depending on the choice of a threshold parameter and so it is considered somewhat case dependent. For these reasons, the Barth–Jesperson limiter was used exclusively for all the results requiring a limiter presented herein.

CHAPTER V

TEMPORAL DISCRETIZATION AND SOLUTION SCHEME

In this chapter, an implicit formulation for the temporal discretization of the equations of fluid motion will be presented and the numerical scheme for a steady–state solution and for an unsteady solution are presented in the two sections thereafter.

5.1 Temporal Discretization

The temporal discretizations are implicit and can be written directly from the integral form of Chapter II using a backward Euler time–differencing scheme as

$$\mathcal{V}\frac{\Delta Q^n}{\Delta \tau} + \sum_{i=1}^{N} \vec{F}(Q^{n+1}) \cdot \hat{n}\, S_i = \sum_{i=1}^{N} \vec{F}^{\nu}(Q^{n+1}) \cdot \hat{n}\, S_i \tag{5.1}$$

where

$$\Delta Q^n = Q^{n+1} - Q^n \tag{5.2}$$

and τ is used to indicate non–dimensional time. The flux vectors are linearized according to

$$\vec{F}^{n+1} = \vec{F}^n + \frac{\partial \vec{F}}{\partial \vec{Q}} \Delta Q^n \tag{5.3}$$

where $\partial \vec{F}/\partial \vec{Q}$ is the flux Jacobian, which is referred to as the A matrix. Using this linearization and rearranging Eq. (5.1) the "delta" form for the following system of linear equations is obtained

$$[A]^n \{\Delta Q\}^n = \{R\}^n \tag{5.4}$$

where

$$[A]^n = \frac{\mathcal{V}}{\Delta \tau} I + \left[\frac{\partial R}{\partial Q}\right]^n \tag{5.5}$$

40

In this equation R is the residual at any time level n, and $\partial R/\partial Q$ indicates the flux Jacobians. For consistency, the same formulation should be used for the fluxes and the flux Jacobians. However, using a higher–order formulation for the Jacobians will result in a larger stencil, leading to an increase in memory and computational time requirements. Hence, only first–order accurate fluxes are used in the Jacobian computations.

The inviscid flux Jacobians are based exclusively on the van Leer FVS since these Jacobians are much cheaper to compute than Roe's FDS Jacobians which are based on the Roe–averaged variables. The analytic flux Jacobians for the 3–D inviscid split–fluxes for dynamic motion of Roe and van Leer are not trivial to derive. In this study, analytic and numeric flux Jacobians for van Leer FVS and numerical flux Jacobians for Roe's FDS are implemented. The analytic Jacobians of the flux vector are presented in the Appendix. The numerical flux Jacobian is computed using difference quotients [62] and the Jacobian elements are obtained by

$$\frac{\partial F_i(Q)}{\partial Q_j} = \frac{F_i(Q + he_j) - F_i(Q)}{h} \tag{5.6}$$

where e_j is the jth unit vector and

$$h \approx \sqrt{machine\ \varepsilon} \tag{5.7}$$

The viscous flux Jacobians are evaluated by differentiating the viscous fluxes based on the finite–volume Galerkin–type approximation as discussed previously in Section 4.1. This formulation results in a central–difference type evaluation for these terms [45].

5.2 Steady–State Solutions

The steady–state solution of the system of equations represented by Eq. (5.4) is obtained with a relaxation scheme whereby $\{\varDelta Q\}^n$ is obtained through

a sequence of iterates, $\{\Delta Q\}^i$, which converges to $\{\Delta Q\}^n$ [45]. To clarify the scheme, $[A]^n$ is written as a linear combination of two matrices representing the diagonal and off–diagonal terms as

$$[A]^n = [D]^n + [O]^n \qquad (5.8)$$

A Jacobi iteration scheme for the solution of the system can be obtained by taking the off–diagonal terms to the right–hand side and evaluate these using the values of $\{\Delta Q\}^i$ from the previous subiteration level i, that is

$$[D]^n\{\Delta Q\}^{i+1} = \left[\{R\}^n - [O]^n\{\Delta Q\}^i\right] \qquad (5.9)$$

To obtain a Gauss–Seidel type relaxation process, with improved convergence rates, a sub–iterative procedure is formulated which uses the latest values of $\{\Delta Q\}$ as they become available. A red–black, or even–odd, approach is taken where the nodes are "colored" according to whether the node is even or odd. All the even–numbered nodes are updated in a subiteration and then all the odd–numbered nodes in the next subiteration, that is

$$[D]^n\{\Delta Q\}^{i+1} = \left[\{R\}^n - [O]^n\{\Delta Q\}^j\right] \qquad (5.10)$$

where i indicates the subiteration level and j corresponds to $i + 1$ for the even–numbered nodes and i for the odd–numbered nodes.

Local time–stepping is used to accelerate the convergence to steady-state. The time–step calculation is based strictly on inviscid stability considerations for each node and is given by

$$\Delta\tau = CFL \frac{\mathcal{V}}{\displaystyle\oint_{\partial\Omega}(|U| + a)dS} \qquad (5.11)$$

In Equation (5.11) $\mathcal{V}$ is the dual–mesh volume, U is the velocity normal to the boundary of the control volume, and a is the speed of sound. Anderson and Bonhaus [45] found computations on 2–D turbulent flow problems remained stable for CFL numbers up to 500. The best convergence they found, however,

on large grids CFL numbers between 100 and 200. In this study, a wide range of CFL numbers for the steady and unsteady solutions of inviscid and viscous problems were used as will be discussed in the results section.

5.3 Unsteady Solutions

For unsteady solutions, the Newton–Relaxation procedure of Whitfield and Taylor [62] was used in this study. In this Newton approach to the procedure, the solution of Equation (5.1) would be perfect if a value for Q^{n+1} could be found that satisfies the system of equations. In other words, find Q^{n+1} such that

$$\mathcal{F}(Q^{n+1}) = 0 \tag{5.12}$$

where

$$\mathcal{F}(Q^{n+1}) = \mathcal{V}\frac{\Delta Q^n}{\Delta \tau} + \sum_{i=1}^{N} F(Q^{n+1}) \cdot \hat{n} S_i - \sum_{i=1}^{N} F^\nu(Q^{n+1}) \cdot \hat{n} S_i \tag{5.13}$$

Newton's method for the function $\mathcal{F}(Q^{n+1})$ would be

$$\mathcal{F}'(Q^{n+1,m})(Q^{n+1,m+1} - Q^{n+1,m}) = -\mathcal{F}(Q^{n+1,m}) \tag{5.14}$$

where $m = 1,2,3,...$ are Newton iterates and $\mathcal{F}'(Q^{n+1})$ is the Jacobian matrix of the vector $\mathcal{F}(Q^{n+1})$. When numerical Jacobians are used the procedure is referred to as Discretized Newton–Relaxation (DNR).

Equation (5.12) uses a first–order approximation for the time–derivative. For higher–order accuracy in time a second–order backward approximation is used which is

$$\frac{\partial Q}{\partial \tau} = \frac{3Q^{n+1} - 4Q^n + Q^{n-1}}{2\Delta \tau} = \frac{3\Delta Q^n - \Delta Q^{n-1}}{2\Delta \tau} \tag{5.15}$$

To obtain a second–order solution, the first–order scheme is run for the first time–step to obtain two time levels for Q.

Time–accurate solutions are obtained by using a constant time–step throughout the computational domain and by converging the Newton iterations. A minimum time–step is determined and then an effective *CFL* is calculated as

$$CFL_{effective} = \frac{\Delta\tau_{actual}}{\Delta\tau_{min\,time-step}} \tag{5.16}$$

In the 3–D unsteady Euler solutions obtained in this study the effective *CFL* ranged from 25 to 500.

For the trajectory simulations it was found that the motor exit condition surface contained the elements with the smallest time–steps. These cells were less than a few per cent of the total number of cells. By allowing these cells to use local time–stepping and letting the rest of the domain to proceed at a higher time–step effective *CFL* numbers on the order of 10 were obtained without significantly affecting the overall time–accuracy of the solution. Details of this technique and its implementation are contained in Chapter VII.

CHAPTER VI

BOUNDARY CONDITIONS

The three types of boundary conditions used in this study were implemented for inviscid, viscous and far–field surfaces. The emulation of a rocket motor exhaust exit surface was handled through a modified far–field boundary condition. For the inviscid and viscous surfaces, during the initial start–ups while using first–order spatial accuracy, a zero pressure gradient condition was also imposed.

6.1 Inviscid Surface

An inviscid surface is impermeable whereby the normal component of velocity is zero while a tangential component is allowed. This is also referred to as a slip condition. For an Euler solution about a maneuvering vehicle, the vehicle body surface is an inviscid surface boundary. Characteristic variable boundary conditions similar to those described in [63] were used to set the values of the flow variables for an inviscid surface. The flow field values on the surface of the vehicle body for pressure, density, and the velocity components are determined by

$$p_{body} = p_{ref} + \rho_{ref} a_{ref} U_{ref} \tag{6.1}$$

$$\rho_{body} = \rho_{ref} + \left(p_{body} - p_{ref}\right)/a_{ref}^2 \tag{6.2}$$

$$u_{body} = u_{ref} - n_x U_{ref} \tag{6.3}$$

$$v_{body} = v_{ref} - n_y U_{ref} \tag{6.4}$$

$$w_{body} = w_{ref} - n_z U_{ref} \tag{6.5}$$

45

where

$$U_{ref} = (u_{ref} - x_t)n_x + (v_{ref} - y_t)n_y + (w_{ref} - z_t)n_z \tag{6.6}$$

The terms with *ref* subscripts are dual mesh cell–averaged and the mesh speed and unit normal terms are face–averaged. That is, characteristic boundary values at the surface are determined from cell (tetrahedron) and boundary surface (triangle) centroid values. The energy at the body surface is calculated using the characteristic values and the equation of state as

$$E_{body} = \frac{p_{body}}{(\gamma - 1)} + \frac{1}{2}\rho_{body}(u_{body}^2 + v_{body}^2 + w_{body}^2) \tag{6.7}$$

6.2 Viscous Surface

A viscous surface corresponds to a no–slip boundary condition whereby the fluid velocity has the same value as the surface velocity. For a non–moving mesh, this surface condition implies the normal and tangential velocity components are zero. For a moving mesh the surface velocities are equal to the mesh speed. In both cases there is a prescribed surface temperature. The enforcement of these conditions is obtained by a modification of the diagonal matrix and right–hand side terms of Equation (5.10) [45]. An expanded representation of one of the rows in Equation (5.10) yields

$$\begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & D_{15} \\ D_{21} & D_{22} & D_{23} & D_{24} & D_{25} \\ D_{31} & D_{32} & D_{33} & D_{34} & D_{35} \\ D_{41} & D_{42} & D_{43} & D_{44} & D_{45} \\ D_{51} & D_{52} & D_{53} & D_{54} & D_{55} \end{bmatrix} \begin{bmatrix} \Delta\rho \\ \Delta\rho u \\ \Delta\rho v \\ \Delta\rho w \\ \Delta E \end{bmatrix} = \begin{bmatrix} RHS_1 \\ RHS_2 \\ RHS_3 \\ RHS_4 \\ RHS_5 \end{bmatrix} \tag{6.8}$$

where *RHS* represents both the residual and the off–diagonal terms on the right–hand side of Equation (5.10) and D_{ij} represents the components of one of the diagonal blocks in [D]. The first row of D_{ij}, to calculate the density from the continuity equation, does not need to be modified and is left unchanged.

First, consider the modification for a non–moving mesh. The second, third, and fourth rows are modified so the solution of Equation (5.10) yields a zero velocity at the nodes on the viscous surface. The fifth row is modified to enforce the prescribed constant wall temperature which is set to the adiabatic wall temperature [26]

$$\frac{T_{wall}}{T_\infty} = 1 + \sqrt{Pr}\frac{\gamma - 1}{2}M_\infty^2 \tag{6.9}$$

The adiabatic wall temperature is used to relate the change in energy at the wall to the change in density which yields

$$\Delta E_{wall} = \frac{T_{wall}}{\gamma(\gamma - 1)}\Delta\rho \tag{6.10}$$

Note this relationship was derived from the non–dimensional form of the state equation. The modified matrix for enforcement of these viscous boundary conditions is

$$\begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & D_{15} \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ -T_{wall}/\gamma(\gamma - 1) & 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \Delta\rho \\ \Delta\rho u \\ \Delta\rho v \\ \Delta\rho w \\ \Delta E \end{bmatrix} = \begin{bmatrix} RHS_1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{6.11}$$

For a moving mesh case, the surface velocity at the boundary nodes are the mesh speed. The relative velocities are zero which gives the no–slip viscous boundary condition. Consider the change in x–momentum

$$\Delta(\rho u)^n = (\rho u)^{n+1} - (\rho u)^n = \rho^{n+1}u^{n+1} - \rho^n u^n = (\Delta\rho^n + \rho^n)u^{n+1} - \rho^n u^n \tag{6.12}$$

or, rearranging

$$\Delta(\rho u)^n - \Delta\rho^n u^{n+1} = \rho^n(u^{n+1} - u^n) \tag{6.13}$$

Since the node velocity is the mesh speed this can be written as

$$\Delta(\rho u)^n - \Delta\rho^n x_t^{n+1} = \rho^n(x_t^{n+1} - x_t^n) = \rho^n\Delta x_t \tag{6.14}$$

This gives relationships for the momentum terms. The adiabatic wall temperature condition, or recovery temperature, uses the relative freestream Mach number. There is a kinetic energy portion of the total energy at the surface due to the mesh (surface) speeds which was zero for the non–moving mesh case. The relationship of Equation (6.10) now includes some additional terms which can be written as

$$\Delta E_{wall} = \frac{T_{wall}}{\gamma(\gamma - 1)}\Delta\rho + \frac{\Delta\rho}{2}(x_t^2 + y_t^2 + z_t^2)^{n+1} + \rho^{n+1}(x_t^{n+1}\Delta x_t + y_t^{n+1}\Delta y_t + z_t^{n+1}\Delta z_t) \qquad (6.15)$$

Strictly speaking, for an implicit formulation the density and mesh speed terms should be at the $n + 1$ time level. Using a linearization for density as was done for the momentum terms the modified matrix for enforcement of the moving mesh viscous boundary conditions, with columns 2 through 5 of D the same as in Equation (6.11), becomes

$$\begin{bmatrix} D_{11} & & & & \\ -x_t^{n+1} & & & & \\ -y_t^{n+1} & & D^2 & D^3 & D^4 & D^5 \\ -z_t^{n+1} & & & & \\ -\frac{T_{wall}}{\gamma(\gamma-1)} - \frac{1}{2}(x_t^2 + y_t^2 + z_t^2)^{n+1} - (x_t^{n+1}\Delta x_t + y_t^{n+1}\Delta y_t + z_t^{n+1}\Delta z_t) & & & & \end{bmatrix} \begin{bmatrix} \Delta\rho \\ \Delta\rho u \\ \Delta\rho v \\ \Delta\rho w \\ \Delta E \end{bmatrix} =$$

$$\begin{bmatrix} RHS_1 \\ \rho^n\Delta x_t \\ \rho^n\Delta y_t \\ \rho^n\Delta z_t \\ \rho^n(x_t^{n+1}\Delta x_t + y_t^{n+1}\Delta y_t + z_t^{n+1}\Delta z_t) \end{bmatrix} \qquad (6.16)$$

For unsteady problems with forced motion, such as an oscillating wing, the surface positions and velocities at the next time–level are known. For a trajectory simulation, coupling the equations of fluid motion with the 6DOF model within a Newton iterate is one way to predict future positions and velocities within a time–step and then correct during the solution process with additional Newton iterates. This would be required if the time scale of the

fluid dynamics is on the same order as those for the rigid body dynamics. In this study, the fluid dynamic time scales were much smaller than the rigid body dynamic time scales and explicit updates of the mesh and surface positions and velocities were used. This will be discussed further in the results section. For a mesh moving with a constant speed, Equation (6.16) reduces to

$$
\begin{bmatrix}
\begin{array}{c} D_{11} \\ -x_t \\ -y_t \\ -z_t \\ -\dfrac{T_{wall}}{\gamma(\gamma-1)} - \frac{1}{2}(x_t^2 + y_t^2 + z_t^2) \end{array} & D^2 & D^3 & D^4 & D^5
\end{bmatrix}
\begin{bmatrix} \Delta\rho \\ \Delta\rho u \\ \Delta\rho v \\ \Delta\rho w \\ \Delta E \end{bmatrix}
=
\begin{bmatrix} RHS_1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
\tag{6.17}
$$

where the superscripts on the mesh speeds were dropped since they are constant and D^j represents columns 2 through 5 which are the same as those in Equation (6.11).

6.3 Far–Field Freestream Surface

As described in Anderson and Bonhaus [45], a far–field freestream surface of the flow field is assumed to be inviscid. Characteristic reconstruction of flow field values needed for the flux computation along this outer boundary can be obtained from two locally one–dimensional Riemann invariants. Many textbooks on compressible flow cover the derivation of the method of characteristics and Riemann invariants ([64], [65]) and only the results as applied in this study will be presented.

The Riemann invariants are constant along characteristic lines defined normal to the far–field freestream surface, the outer boundary, and are given by

$$
R^{\pm} = U \pm \frac{2a}{\gamma - 1}
\tag{6.18}
$$

where the plus and minus sign correspond to right– and left–running characteristics, or sound waves, respectively. The contravariant velocity, U, takes into account the mesh speed. Recall from Chapter II the form of U is

$$\vec{V} \cdot \hat{n} = U = (u - x_t)n_x + (v - y_t)n_y + (w - z_t)n_z \qquad (6.19)$$

For subsonic flow, R^- is evaluated from freestream conditions outside the computational domain and R^+ is evaluated from present values of the flow field data at each node along the boundary which is inside the computational domain. For supersonic outflow, R^- is evaluated the same way as R^+ for subsonic flow – from interior data. For supersonic inflow, R^- and R^+ are evaluated using freestream conditions from outside the computational domain. The normal velocity and speed of sound local to the far–field boundary surface are determined using the Riemann invariants as

$$U_{boundary} = \tfrac{1}{2}(R^+ + R^-) \qquad (6.20)$$

$$a_{boundary} = \frac{\gamma - 1}{4}(R^+ - R^-) \qquad (6.21)$$

If $U_{boundary}$ is greater than zero there exists an outflow condition and flow field reference variables (subscript *ref*) are taken from inside the computational domain. If $U_{boundary}$ is less than zero there exists an inflow condition and flow field reference variables are taken from freestream conditions. The velocity components are evaluated by decomposing the normal and tangential velocity vectors as

$$
\begin{aligned}
u_{boundary} &= u_{ref} + n_x\left(U_{boundary} - U_{ref}\right) \\
v_{boundary} &= v_{ref} + n_y\left(U_{boundary} - U_{ref}\right) \\
w_{boundary} &= w_{ref} + n_z\left(U_{boundary} - U_{ref}\right)
\end{aligned}
\qquad (6.22)
$$

The entropy on the far–field boundary surface is determined as

$$S_{boundary} = \frac{a_{ref}^2}{\gamma \rho_{ref}^{\gamma - 1}} \qquad (6.23)$$

Then the density on the far–field boundary surface can be determined from the entropy and the speed of sound as

$$\rho_{boundary} = \left(\frac{a^2_{boundary}}{\gamma s_{boundary}} \right)^{\frac{1}{\gamma-1}} \tag{6.24}$$

Finally, the pressure and the energy are calculated from the equation of state given in Chapter II and the flux evaluated using these characteristic reconstruction values.

6.4 Far–Field Motor Exit Surface

A simple emulation of a motor exit boundary surface works in essentially the same way as a far–field boundary surface. An exit pressure, density, and Mach number are assigned to these surfaces. In this study, chemical reactions are neglected in the exhaust stream so standard air is assumed. These exit plane values are non–dimensionalized using freestream values in a consistent manner as the rest of the flow field values. An example for a solid rocket motor used in a typical advanced air–to–air missile will be presented in the next chapter.

CHAPTER VII

RESULTS

Recently, a special section in the May 1998 AIAA Journal was devoted to assessing the credibility of CFD simulations. According to the summary therein, written by Mehta [66], "validation assesses whether correct things are performed, and verification assesses whether they are performed correctly." The results presented in this chapter constitute the verification and validation (V&V) of the 3–D unstructured CFD method for dynamic motion documented herein.

An accepted method for validation of the CFD solutions generated by a particular method is comparison to CFD–quality wind–tunnel data. This type of data provides more detail than integrated wind–tunnel data and is more suitable for code validation. For instance, a favorable comparison of the prediction of the pressure distribution along a wing with experimental data for a transonic Mach number at low angle of attack with no separation is probably a reasonable indicator that the Euler flow solver is correct. For this same configuration, a Navier–Stokes solver should give the same pressure distribution results as well as predicting viscous data such as skin friction distributions.

For steady–state solutions, there is an abundance of CFD–quality wind–tunnel data available. For unsteady solutions, the wind–tunnel data available is restricted to tunnel models that are not free flying – oscillating configurations or, for a trajectory simulation, quasi–steady data. Most of the unsteady data, however, is not CFD–quality data in that integrated results

are measured and presented such as lift and drag coefficient time histories. A favorable comparison of an unsteady CFD solution to an integrated result really does not assess the detailed time–accuracy of the solution. The integrated result may fortuitously match the experimental data while grossly missing the flow field distributions at a particular instance. The unsteady wind–tunnel data used herein was selected because it was specifically designed to provide CFD–quality data for time–accurate code validation.

As previously mentioned, for trajectory simulations the wind–tunnel data is restricted to non–free flying models. Flight test data for a true trajectory simulation is usually not CFD–quality for validation in that integrated results are presented due to limitations in real–time measurement of flow field data. Hence, the validation results for the equations of fluid motion in the flow solver of the 3–D unstructured CFD method for dynamic motion uses wind–tunnel data. The validity of the results for the trajectory simulation, which couples in the 6DOF model, appeals to heuristic arguments. A summary of the results to be presented in the following sections are as follows.

First, the far–field boundary condition for steady relative motion is validated with a cube in arbitrary motion. Next, the inviscid boundary condition is validated with wind–tunnel data comparisons for a delta and a rectangular wing at various transonic Mach numbers. The viscous boundary condition is validated for steady relative motion with wind–tunnel data for a waisted–body of revolution. The unsteady inviscid dynamic motion validation is assessed with time–accurate wind–tunnel data for an oscillating rectangular wing. Lastly, an unsteady inviscid trajectory simulation for a missile with a motor condition is presented. As mentioned above, the missile solutions will have to appeal to heuristic arguments due to the lack of CFD–quality data.

7.1 Arbitrary Motion Cube

A uniform freestream is a valid solution of the equations of fluid motion. Furthermore, an arbitrary moving mesh should not affect the flow field. Hence, the first results presented to validate dynamic motion are those for a cube moving through a uniform freestream.

Figure 7.1 shows the surface mesh and orientation for the arbitrary motion cube. The 3–D mesh of consists of 7,328 nodes with 35,443 tetrahedrons and with 5,360 faces on the far–field boundary surface. The mesh was moved arbitrarily in a uniform flow field at $M_\infty = 2.0$ for 200 iterations total – 100 iterations each of first– and second–order spatial accuracy with local time–stepping, a *CFL* of 20.0 and with Roe's FDS and van Leer's FVS. There were no changes to the uniform flow field above machine accuracy in the numerical solution. Thus, the far–field surface boundary condition was considered validated.

(a) Unstructured Mesh   (b) Cube Orientation
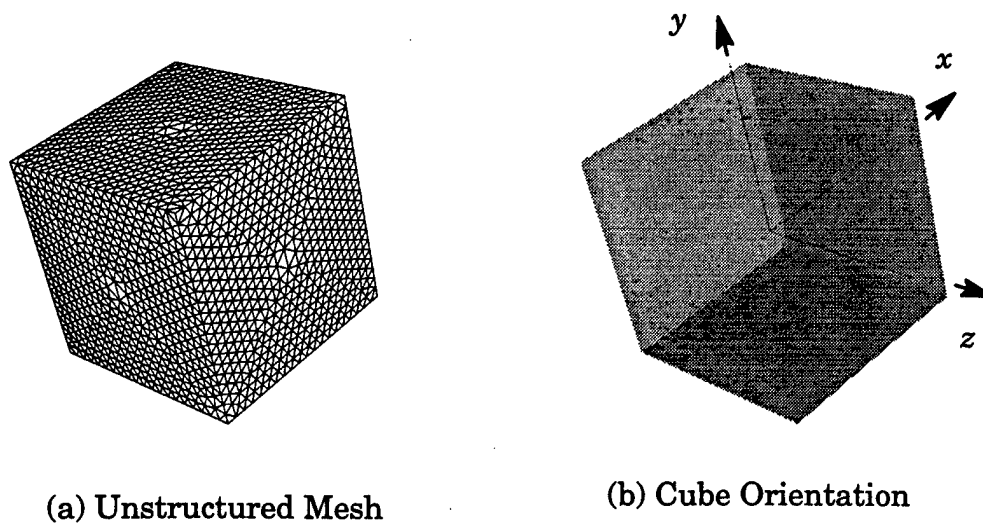
Figure 7.1 Unstructured mesh (a) and cube orientation (b)

7.2 Steady Solutions

Full 3–D Euler and Navier–Stokes steady–state CFD solutions compared to CFD–quality wind tunnel data were used to assess the accuracy of the flow solver. The data used for comparison consisted of pressure distributions for the inviscid results and pressure and skin friction distributions for the viscous results. Then, steady relative motion was applied to the same problems to evaluate the inviscid and viscous surface boundary conditions for a moving mesh. If the solutions obtained for steady relative motion were the same as the solutions without mesh motion then these boundary conditions could be considered valid.

There were two 3–D wing configurations used for the inviscid solutions. A clipped delta wing at transonic Mach numbers of 0.852 and 0.95 and angles of attack (AOA) of zero and two degrees were used to assess relative motion with mesh speeds in the x and y directions. A rectangular wing at Mach = 0.8 and zero degrees AOA further validated the inviscid surface boundary condition and provided the starting solution for the unsteady oscillating wing solutions. The configuration for the viscous steady relative motion solutions was a waisted–body of revolution at a Mach number of 1.4 and zero degrees AOA. For this case, the wind tunnel data included skin friction data as well as pressure coefficient data which provided a more complete assessment of this boundary condition.

7.2.1 Delta Wing Euler Solutions

The solutions presented in this section were used to validate the inviscid surface boundary condition for steady relative motion. The surface pres-

sure coefficient data was obtained from an Arnold Engineering Development Center (AEDC) wind–tunnel test of a clipped delta wing (NACA 64A010 airfoil section) at transonic Mach numbers of 0.852 and 0.95 and angles of attack of zero and 2 degrees [71].

Since yaw angles were not initially being considered, and to save on computational resources, the computational domain was modelled using a symmetry plane with a slightly rectangular far–field outer boundary placed at 5 wing–root chords away from the wing surface in each direction. The unstructured mesh for the 3–D Euler solutions was 42,584 nodes with 16,376 surface triangles and a total of 228,984 tetrahedrons. The nodes were clustered along the leading edge which implicitly provided a slight clustering along the span. A uniform node distribution was used along the trailing edge which, in retrospect, should probably have been clustered also. The distributions were satisfactory for evaluating the inviscid boundary condition. Figure 7.2 shows the symmetry plane and the wing surface node distributions for the NACA 64A010 clipped delta wing.

(a) Symmetry Plane   (b) Wing Surface Mesh
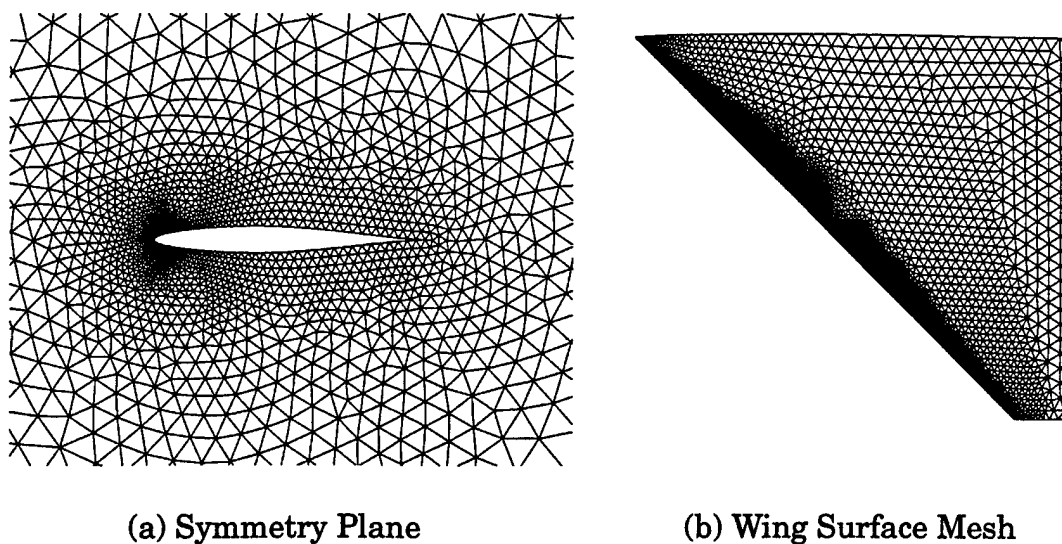
Figure 7.2 Clipped delta wing symmetry plane (a) and surface mesh (b)

The following steady solutions were run with local time–stepping for 2000 iterations with Roe's FDS and a CFL = 3.0. The first 200 iterations used first–order spatial accuracy and a zero pressure gradient surface boundary condition to get past the initial transients. After the first 200 iterations, the spatial accuracy was switched to higher–order and characteristic variable boundary conditions were used. The Barth–Jesperson limiter and Gram–Schmidt least–squares linear reconstruction were used for higher–order accuracy as discussed previously.

For all the Euler solutions to be presented the root mean square (RMS) of the density residual and the lift coefficient were used to monitor the convergence of the numerical solution. The density residual is used because it is the most sensitive of the flow field variables. The lift coefficient is monitored because when it settles down when the pressure distribution about the body has stabilized. The $\log_{10}$ of the RMS of the density residual indicates the number of orders of magnitude reduction of the density residual. Typically, a three to four order magnitude reduction from the initial residual is considered converged for engineering applications. With the Barth–Jesperson limiter the residual reaches a steady cyclic state. As mentioned by Venkatakrishnan [5], the Barth–Jesperson multidimensional limiter can be thought of as a generalization of a min–mod limiter and, as such, can have convergence difficulties. The Barth–Jesperson limiter was needed for the higher Mach number and AOA cases so it was used for all the solutions for the delta wing for consistency. As will be shown in later sections, for solutions run without the limiter, the convergence goes to machine zero.

Figure 7.3 is a comparison of the Euler solution pressure coefficient distribution for Mach 0.852 and zero degrees AOA with the AEDC experimental data at a wing butt line (BL) of 8.3. This corresponds to approximately 64% semispan (8.3/13.0 in inches). The symmetry of the numerical solution about this symmetrical airfoil cross–section is shown by plotting the upper and lower surface results for comparison with the upper and lower surface AEDC experimental data. Despite the lack of an experimental data point near the leading edge, it appears the leading edge expansion was resolved quite nicely. The Euler predictions in the supersonic region also compare favorably to the experimental data. The experimental data and the CFD results at semispan locations closer to the wing root had stronger shocks. The CFD results predicted the location of the shocks slightly aft of the experimental data which is as expected for an Euler solution.
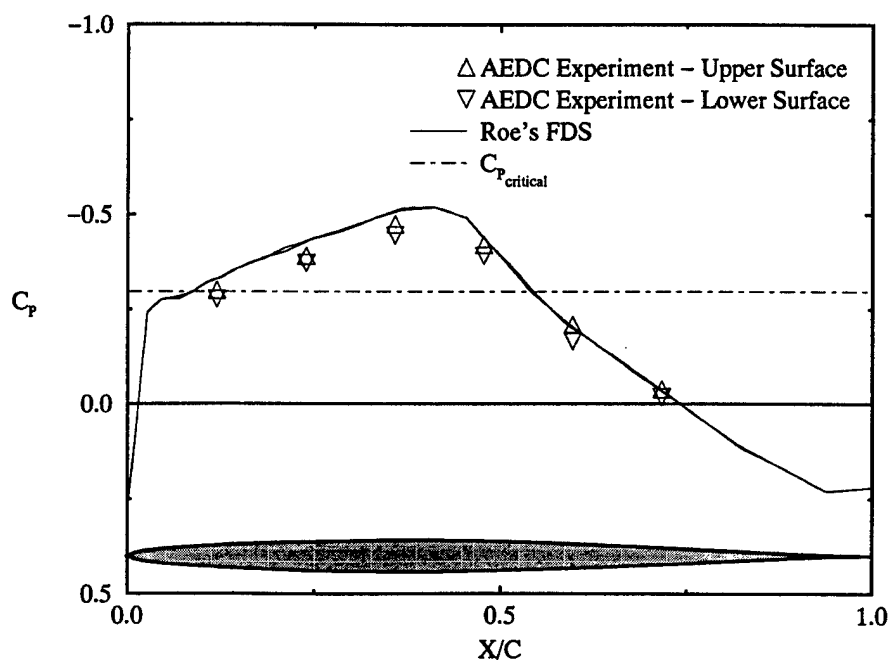
Figure 7.3 Delta wing pressure coefficient distribution at Mach = 0.852 and AOA = 0 at wing BL = 8.3 for steady Euler solution with no mesh motion
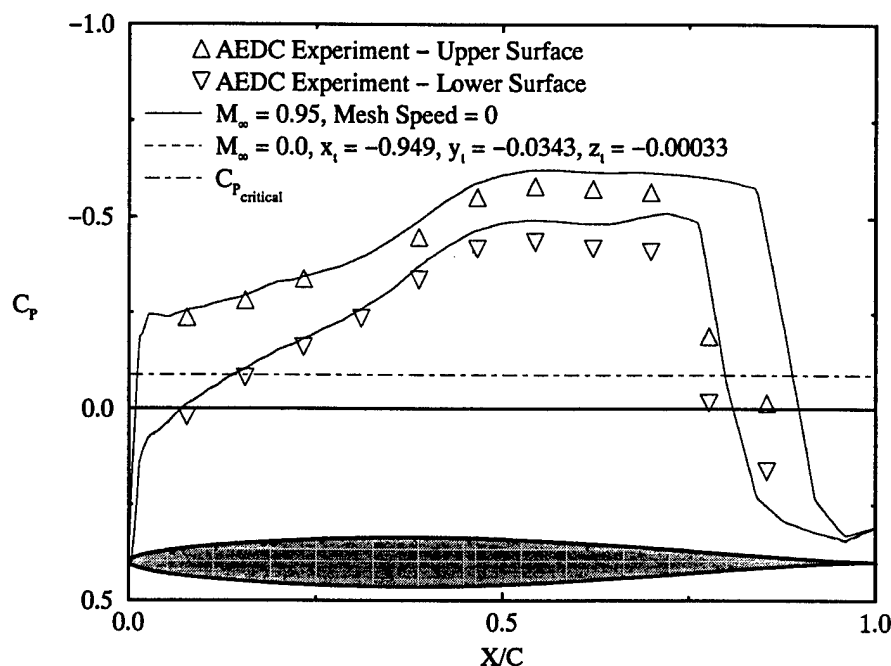
59
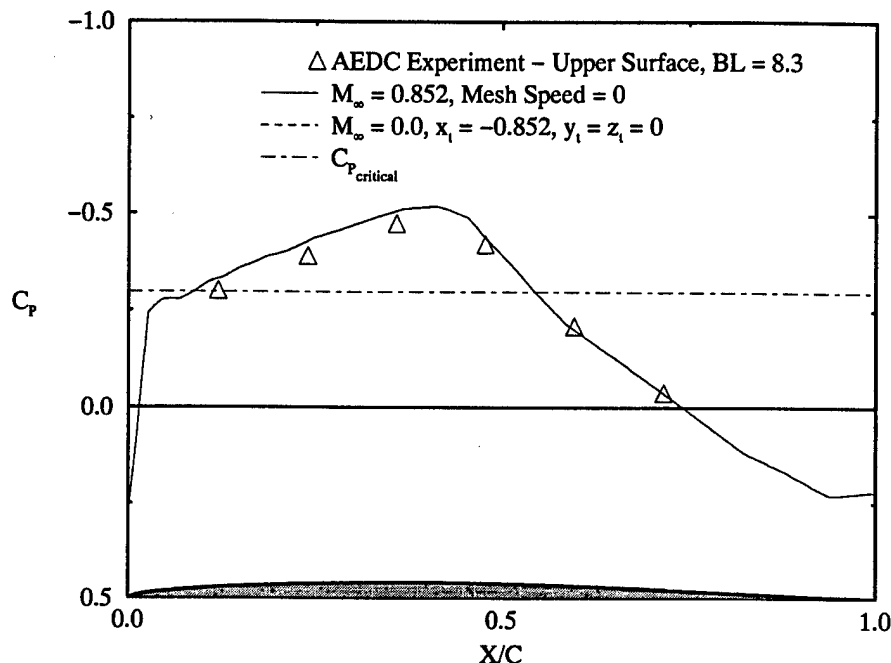
Figure 7.4 Delta wing pressure coefficient distribution at Mach = 0.852 and AOA = 0 at wing BL = 8.3 for steady Euler solution with and without mesh motion

Figure 7.4 shows upper surface pressure coefficient comparisons between the AEDC data, the previous steady Euler solution with no mesh motion, and a steady relative motion Euler solution where the delta wing and rigidly attached mesh move at Mach = −0.852 into a quiescent flow field. Hence, the relative Mach numbers for the two Euler solutions are the same. These line plots of the predicted pressure distributions for both Euler solutions are virtually identical. This gives confidence the inviscid surface boundary condition is being modelled correctly for mesh motion in the x direction. The next case will assesss the validity of the inviscid surface boundary condition for mesh motion in the x, y, and z directions simultaneously. Before looking at those results a consideration of the iteration history proves quite interesting.
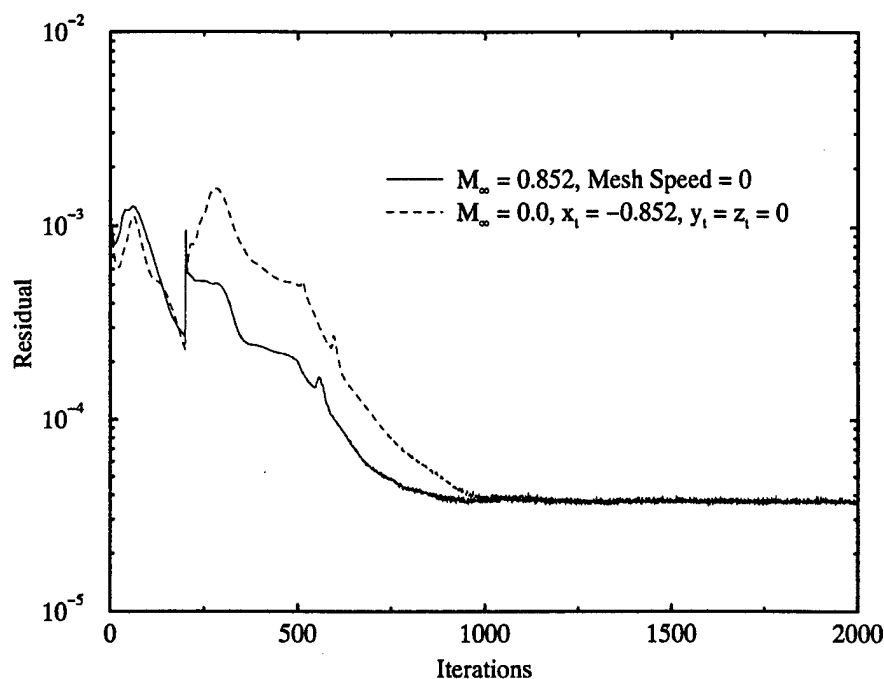
Figure 7.5 Delta wing density residual history comparisons

Figure 7.5 presents a comparison of the density residual history for the delta wing Euler solutions with and without mesh movement. The jump in the residual when the higher–order spatial accuracy is switched on at 200 iterations is clearly evident. Both solutions were converged after a 1000 iterations. The transient history of the solutions are not the same because the flow fields start with different initial freestream conditions. When there is no mesh movement every node is initialized with final freestream total conditions. For the mesh movement case the initial freestream conditions of every node are lower because of the quiescent state. That means the total energy of the field is greater in the first case and the flow field adjusts to the influence of the body quicker. Since an unsteady simulations can start moving from a steady solution on a stationary mesh, it makes sense to obtain the starting solution on a stationary mesh and then start the mesh movement.

Figure 7.6 Delta wing pressure coefficient distribution at Mach = 0.95 and AOA = 2.07 degrees at wing BL = 4.7 for steady Euler solutions with and without mesh motion

Figure 7.6 presents pressure coefficient comparisons at Mach = 0.95 and AOA = 2.07 degrees on the upper and lower surfaces of the clipped delta wing between the AEDC data and the Euler solutions with and without mesh movement at a wing BL = 4.7 – semispan location of around 36% (4.7/13 inches). At this Mach number and semispan location the flow is predominantly supersonic with a strong shock near the trailing edge. Both the moving and nonmoving mesh solutions are virtually identical and slightly overpredict the expansion and shock location. These are typical of Euler solutions and mesh refinement would not improve the comparisons to the experimental data. The important point is these results seem to indicate the inviscid surface boundary condition for a moving mesh calculation is working properly.

7.2.2 Rectangular Wing Euler Solutions

The solutions presented in this section were generated to compare to the steady pressure measurements taken on a half–model of a rectangular wing of aspect ratio 4 from an AGARD wind–tunnel experiment described in reference [67]. These solutions will be used as initial conditions for the un-steady results to be shown in a later section. The wing was mounted on a half–body attached to a sidewall of the Royal Aeronautical Establishment (RAE) 8 x 8 ft. wind tunnel. The mounting was done with the intent of dis-placing the wing model far enough away from the tunnel wall to be outside the wall's boundary layer. The rectangular wing cross section is a NACA 64A010 airfoil with a thickness/chord ratio modified to 10.6%.

A series of uniformly spaced unstructured meshes were generated to study the effect of mesh density on the accuracy of a steady–state Euler solu-tion. This was also used to determine which spacing to use for the unsteady pitching wing results. These high–quality meshes were generated using the advancing–front/local–reconnection (aflr3) procedure of Marcum described in references [68] to [70].

Figures 7.7 to 7.9 show the node point distribution on the symmetry plane with uniform spacings about the wing surface of 0.02c, 0.025c and 0.03c, respectively. Note that c stands for wing chord. Figures 7.10 to 7.12 show the node point distribution for the far–field boundaries as viewed from the inside and the outside. The far–field boundary was modeled as a half–sphere with a radius of 7 chord lengths. The total number of node points for the three grids were 57,741, 39,320 and 28,624, respectively. This gave a total number of tet-rahedrons of 300,912, 205,877, and 149,759, respectively. The boundary face totals came to 27,348, 18,652 and 13,614, respectively.
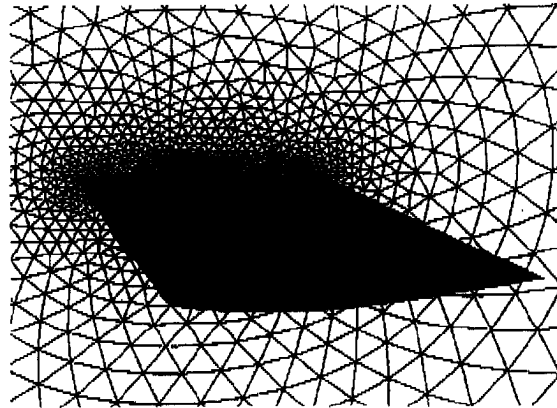
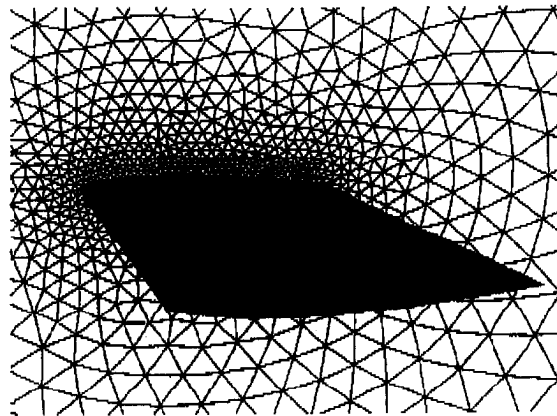**Figure 7.7 Rectangular wing and symmetry plane, uniform spacing = 0.02c**

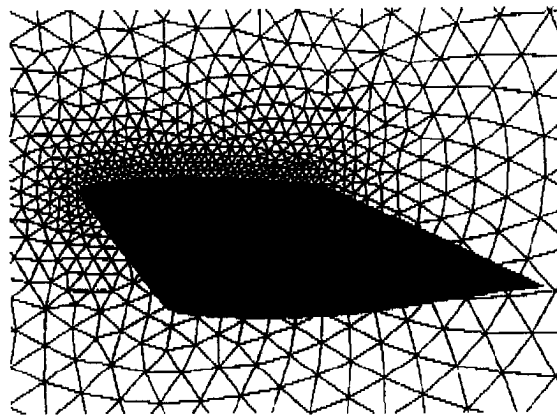**Figure 7.8 Rectangular wing and symmetry plane, uniform spacing = 0.025c**

Figure 7.9 Rectangular wing and symmetry plane, uniform spacing = 0.03c
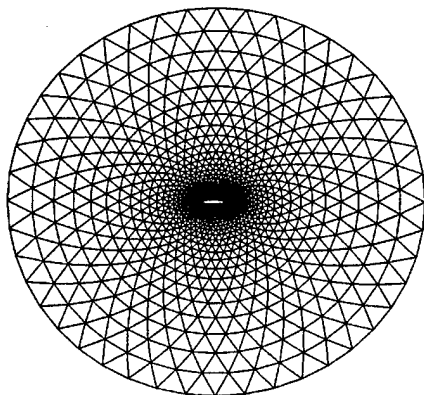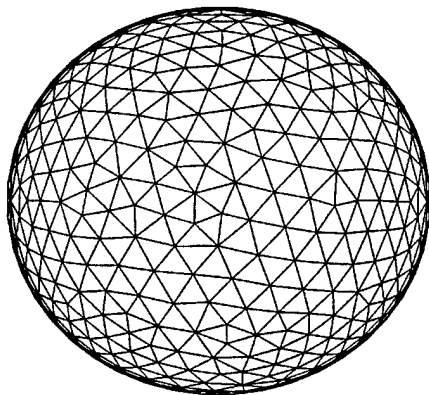
Figure 7.10 View from far–field to symmetry plane

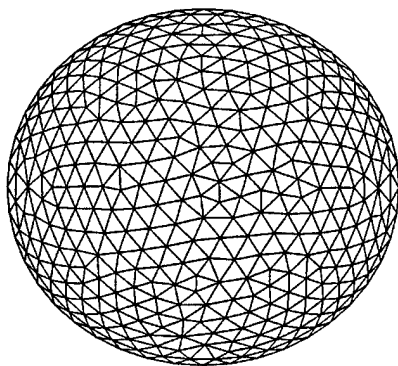Figure 7.11 View from outside far–field to rectangular wing

Figure 7.12 View from rectangular wing to far–field

The following solutions were run for 2,000 iterations using Roe's FDS, local time–stepping for accelerated convergence, and the higher–order spatial accuracy was switched on after the first 200 iterations. Convergence was achieved after about 1,000 iterations with a reduction in the density residual by three orders of magnitude with no noticeable change in the density residual after another 1,000 iterations.

Figures 7.13 to 7.15 show upper surface pressure coefficient comparisons between steady Euler solutions on the three meshes and the RAE experimental data for $M_\infty = 0.8$ and AOA = 0 degrees for 50, 77 and 94% semispan locations, respectively. In all three figures the inadequacy of the mesh resolution at the leading edge is readily apparent as the leading edge expansion is over predicted. At 50%, semispan, Figure 7.13, the mesh with 0.02c uniform spacing resolves the suction peak and shock better than the other two meshes. At the 77 and 94% semispan locations, disregarding the leading edge region, all three grids are in close agreement with each other. This is because the suction peak and shock strength further out towards the wing tip is weaker.
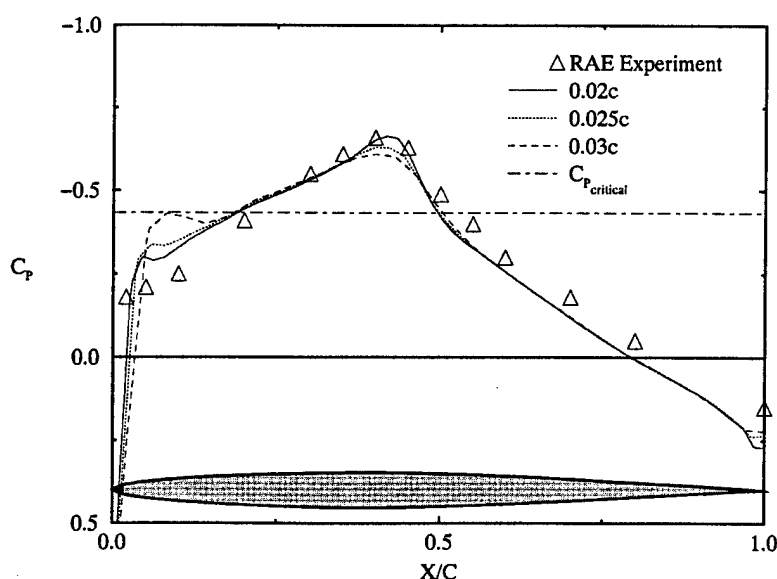
Figure 7.13 Rectangular wing surface pressure coefficient, semispan = 50%
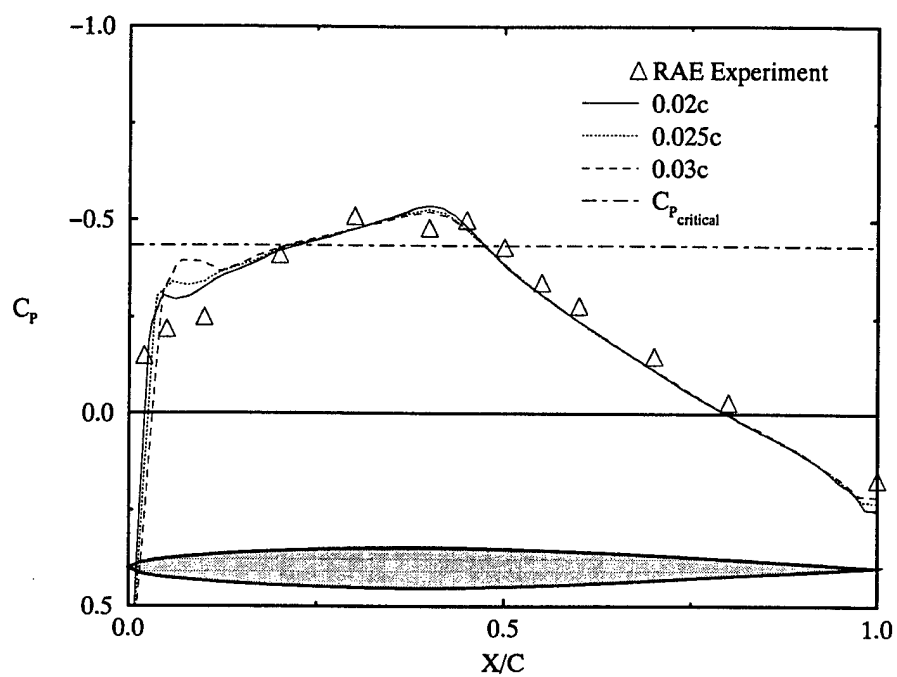
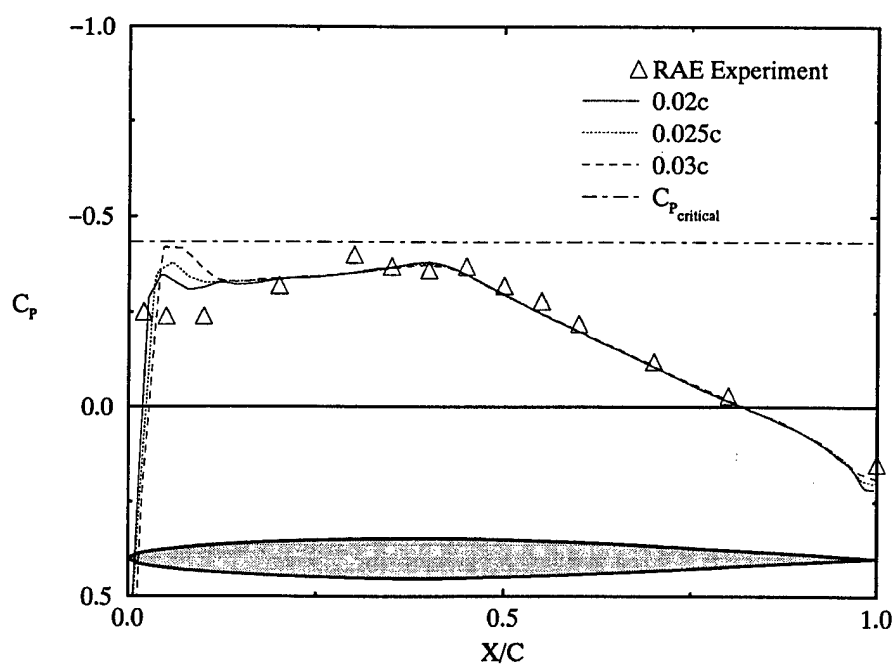Figure 7.14 Rectangular wing surface pressure coefficient, semispan = 77%

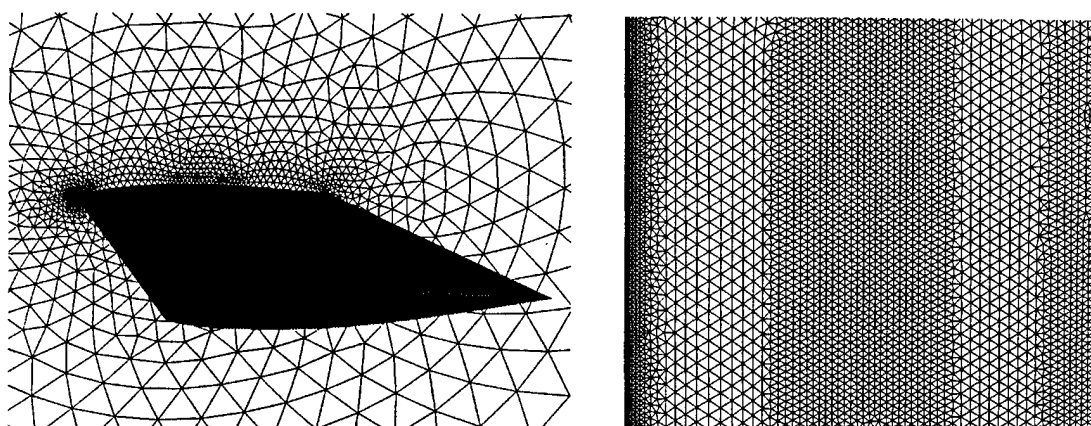Figure 7.15 Rectangular wing surface pressure coefficient, semispan = 94%

Figure 7.16 Rectangular wing and symmetry plane, refined mesh spacing

Figure 7.16 shows the symmetry plane and a close–up of the wing sur-face for the refined mesh generated from these results which was constructed with additional nodes about the leading and trailing edges and in the mid–chord region. It consists of 96,755 nodes and 518,189 tetrahedrons.

Figures 7.17 to 7.19 show comparisons between the refined and the 0.02c mesh and steady Euler predictions and RAE experiment data for $M_\infty = 0.8$ and AOA = 0 degrees for 50, 77 and 94% semispan, respectively. In all three figures the leading edge expansion is resolved much better and compares well with the experimental data for the refined mesh. Note the sharper shock resolution of the refined mesh Euler solution for 50% semispan, Figure 7.17, which does not match the experimental data as well as the 0.02c mesh. This is due to the viscous boundary layer in the data which smears the true shock. The coarser 0.02c mesh also smears the Euler solution while the refined mesh Euler solution produces a sharper shock. These differences will come up again in the unsteady results. For the refined mesh solution at 77 and 94% semispan, Figures 7.18 and 7.19, the results match the experimental data better than the 0.02 c mesh. The refined mesh was selected for use in the unsteady computations and validations to be presented in the next section.
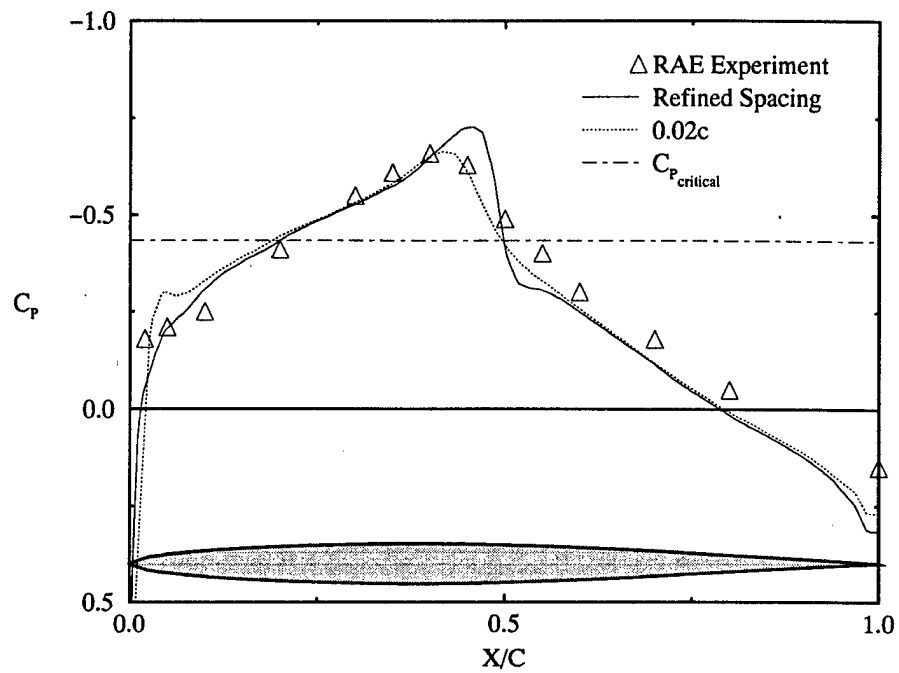
Figure 7.17 Rectangular wing surface pressure coefficient, semispan = 50%
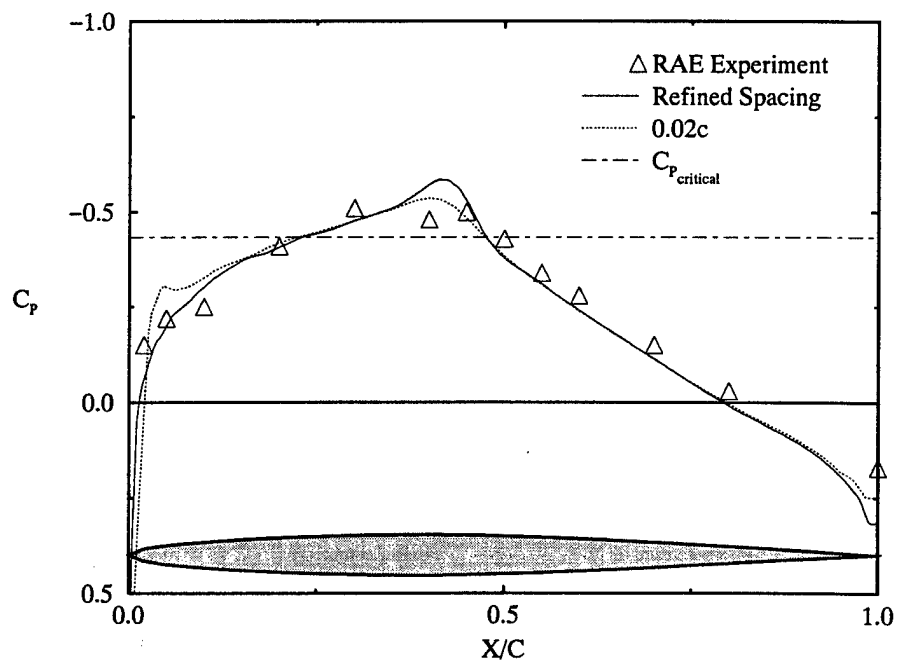

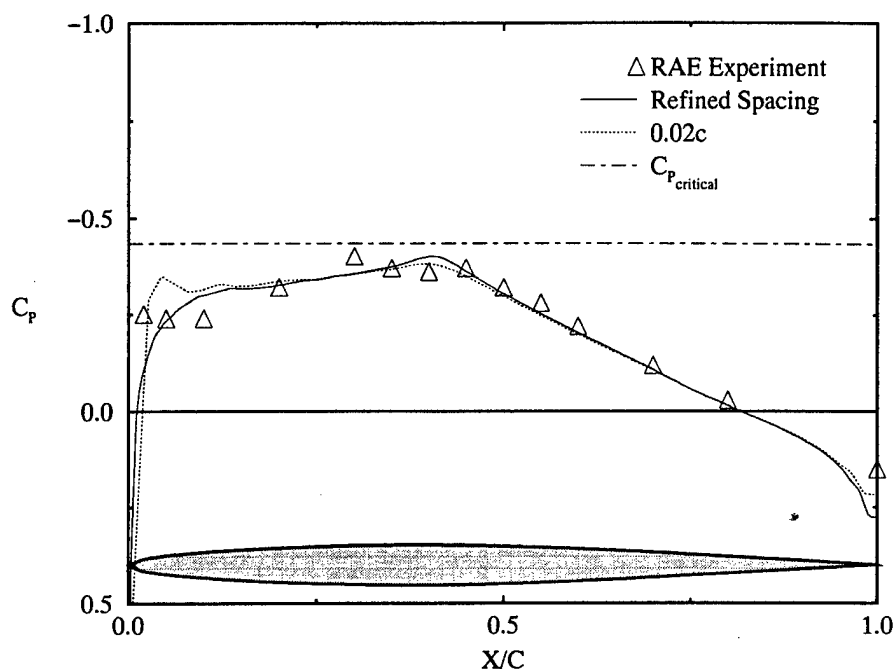Figure 7.18 Rectangular wing surface pressure coefficient, semispan = 77%

Figure 7.19 Rectangular wing surface pressure coefficient, semispan = 94%

Before proceeding to the next section a brief discussion of the reasons for selecting Roe's FDS over van Leer's FVS and the linear reconstruction results are presented. Figure 7.20 presents an upper surface pressure coefficient comparison between Roe's FDS and van Leer's FVS for steady Euler solutions and the RAE experimental data at Mach = 0.8 and AOA = 0 degrees for the 0.02c uniform spaced mesh at semispan = 50%. The results from Roe's FDS matches the experimental data better in the leading edge expansion region and in the vicinity of the shock. These results were typical and Roe's FDS was used consistently throughout this study. Figure 7.21 presents a similar comparison using Roe's FDS and the two types of techniques for linear reconstruction − Green's theorem and Gram–Schmidt least–squares. In this case the results are virtually identical and the Gram–Schmidt process was used.
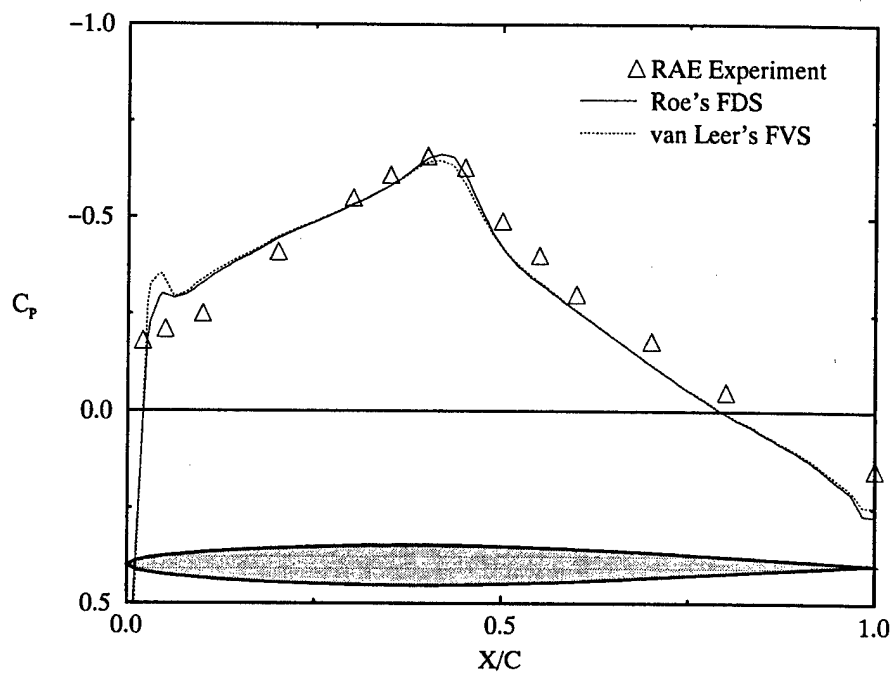
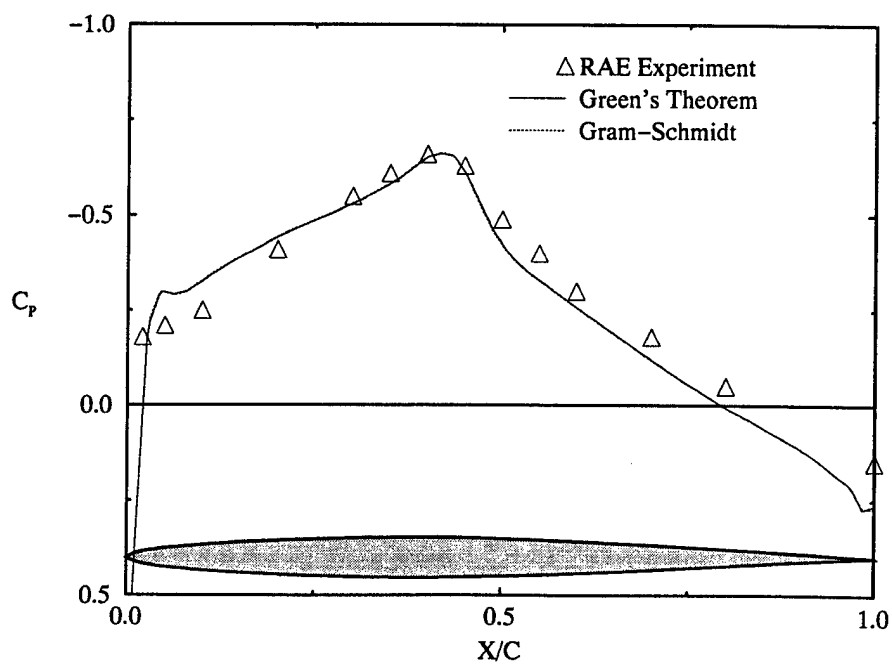Figure 7.20 Roe's FDS and van Leer's FVS at semispan = 50%

Figure 7.21 Green's theorem and Gram–Schmidt at semispan = 50%

7.2.3 Waisted–Body Navier–Stokes Solutions

The viscous solutions presented in this section were generated to compare to the steady pressure and local skin fiction measurements taken on a waisted–body of revolution from a wind–tunnel experiment conducted by collaboration between the Aerodynamische Versuchsanstalt and the Royal Aircraft Establishment described in reference [73]. The model was mounted on a sting and tested in the RAE 8 x 8 ft. wind tunnel at Mach numbers between 0.6 and 2.8 and Reynolds numbers, based on body length, between 5 and 20 million. The model was designed to produce axisymmetric converging flow with an adverse pressure gradient. This was done by starting from the nose with a parabolic body with concavity over the aft end and adding a flare to obtain diverging flow while retaining some of the adverse pressure gradient. Figure 7.22 shows the wind tunnel model. It was not necessary to model
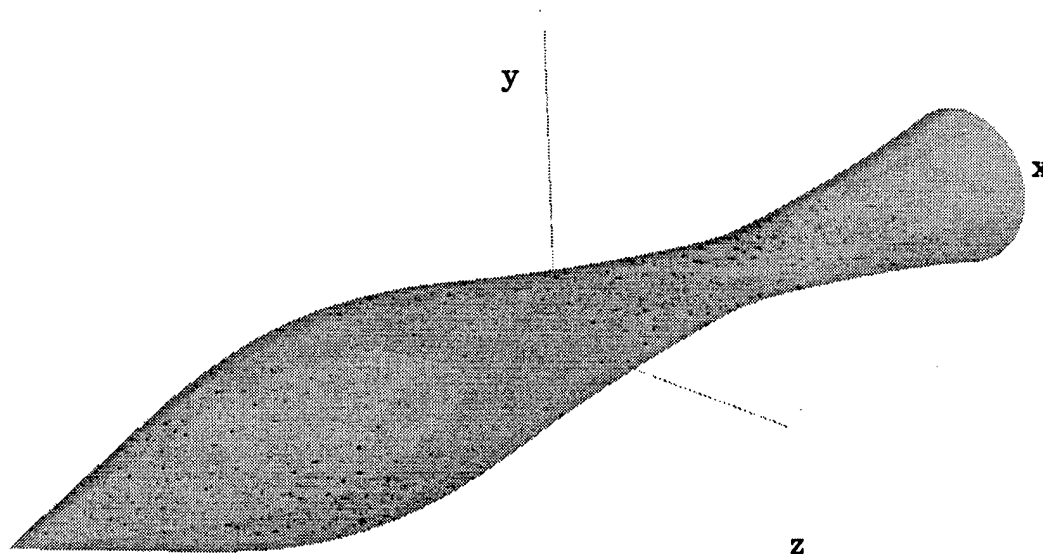

Figure 7.22 Waisted–body of revolution geometry and axes orientation

A full 3–D unstructured mesh was generated which consisted of 165,187 nodes, 17,544 body surface triangles and a total of 957,580 tetrahedrons. The wind tunnel model body length was 60 inches. The numerical mesh model body length was nondimensionalized to 1 and the outer far–field boundary was positioned 5 body lengths from the surface to form a sphere of diameter 10. The mesh generation method, Marcum's *aflr3* [68] – [70], uses advancing–front/advancing–normal point placement with direct insertion and iterative local–reconnection to efficiently generate high–quality meshes suitable for viscous flow applications [74]. For the supersonic case considered with flow field conditions of $M_\infty = 1.4$, AOA = 0 degrees, and Reynolds number of 10.2×10^6, the node/mesh spacing off the wall was set to 1×10^{-6} which gave computed y^+ values over the body surface between 1 and 3.

For the wind tunnel experiment a transition trip was attached on the nose of the body at X/L = 0.025. There were 29 pressure taps located along two axial rows at circumferential angles, φ, of 0 and 30 degrees. The hole for the pressure tap at X/L = 0.15 became blocked early in the test matrix and could not be cleared. The skin friction was measured by a razor blade technique at the $\varphi = 0$ degrees. See reference [73] for additional details concerning the tunnel model geometry and the test set–up.

The Navier–Stokes solutions were run for 1000 iterations using Roe's FDS, local time–stepping, a *CFL* of 1.0, the Barth–Jesperson limiter, the Spalart–Allmaras turbulent model with fully turbulent flow assumed from the nose at X/L = 0 and the higher–order spatial accuracy was switched on after the first 100 iterations. Then the *CFL* was increased to 5.0 and the solutions run another 4000 iterations. The solution was fairly converged around 2000 iterations, but the pressures in the waist region had not quite settled down.

Figure 7.23 presents pressure coefficient comparisons between the RAE experimental data and the steady Navier–Stokes solution at 1,500 and 5,000 iterations during the solution process. The experimental data at $\varphi = 0$ degrees is represented with an upright triangle and will be used throughout. At 1,500 iterations the flow solver has captured the pressure distribution in the zero and favorable pressure gradient regions (X/L = 0.0 to 0.142 and 0.142 to 0.4, respectively) fairly well while the adverse pressure gradient region (X/L = 0.4 to 1.0) needs more iterations to settle down. By 5,000 iterations the solution is converged. This is reflected in the iteration history for the density residual and the lift coefficient shown in Figures 7.24 and 7.25. For this case, the limiter was needed to handle the base region flow transients. The use of the limiter did not unduly influence the overall convergence.
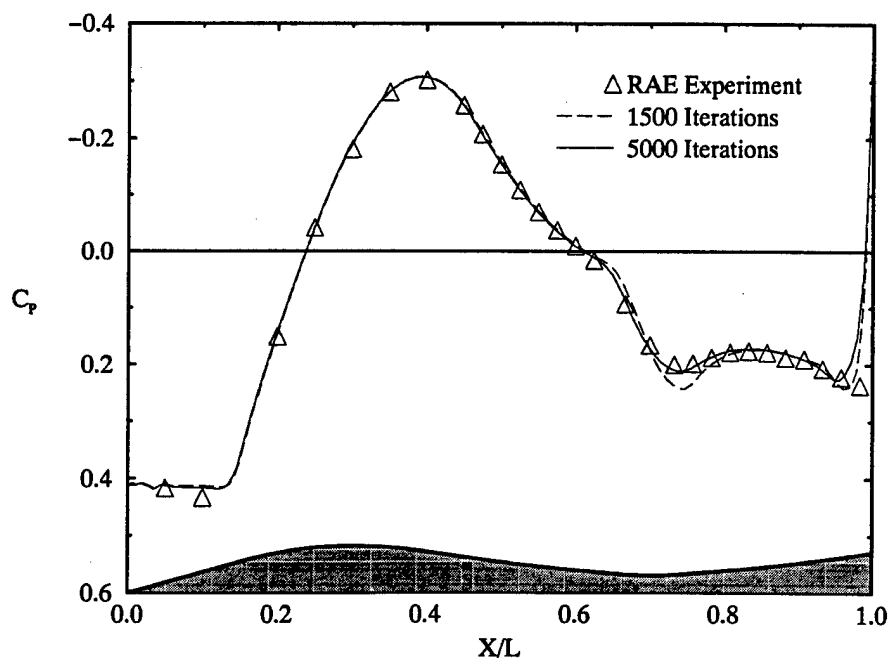

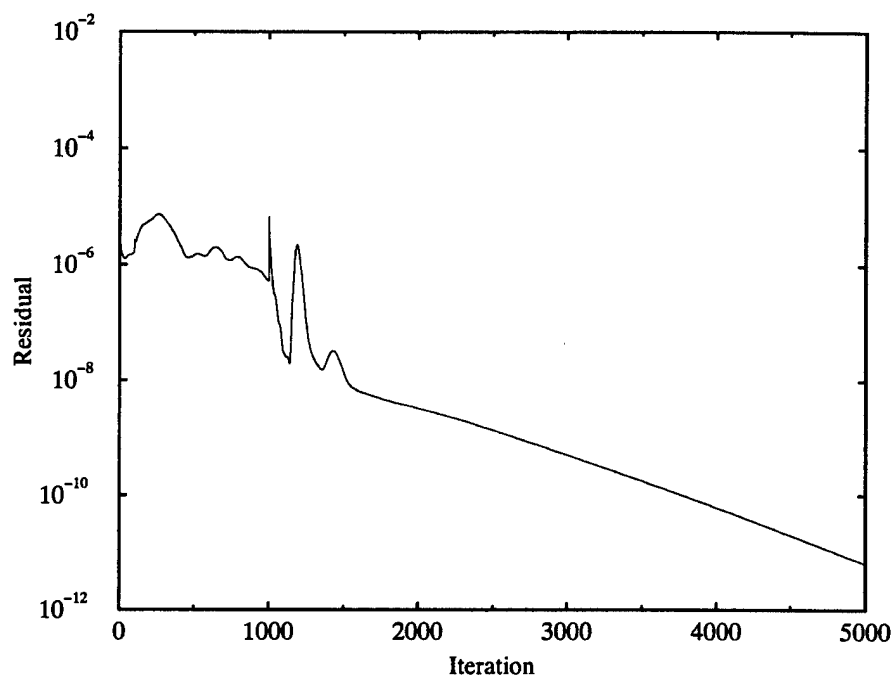Figure 7.23 Waisted–body pressure coefficient comparisons at Mach = 1.4

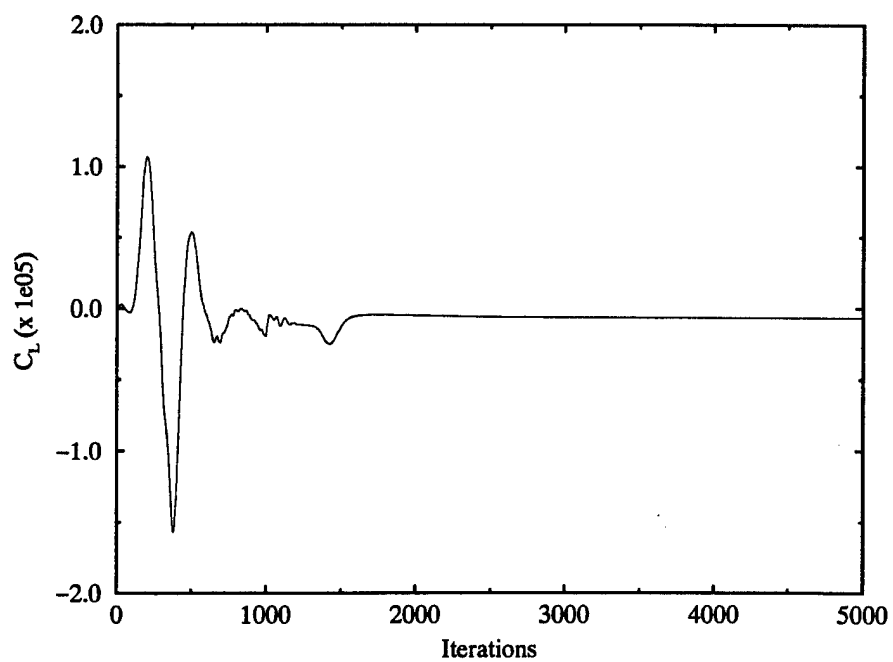Figure 7.24 Waisted–body density residual history at Mach = 1.4

Figure 7.25 Waisted–body lift coefficient history at Mach = 1.4

Figure 7.26 presents skin friction coefficient comparisons between the RAE experimental data and the steady Navier–Stokes solution at 5,000 iterations. A scatter plot showing the computed skin friction coefficient at every surface node was used to show the symmetry of the numerical results. There is excellent qualitative agreement between the experimental data and the Navier–Stokes results. At X/L = 0.142, the end of the conical nose section, there is discontinuity in the model surface curvature which appears to cause a slight separation and may account for the blockage of the pressure tap at X/L = 0.15. The flow solver appears to sense this phenomenon. In the converging flow section, from X/L = 0.4 to 0.7 the adverse pressure gradient causes a thickening of the boundary layer with maximum thickness and minimum skin friction coefficient near the waist. The diverging flow after the waist thins the boundary layer and increases the skin friction coefficient.
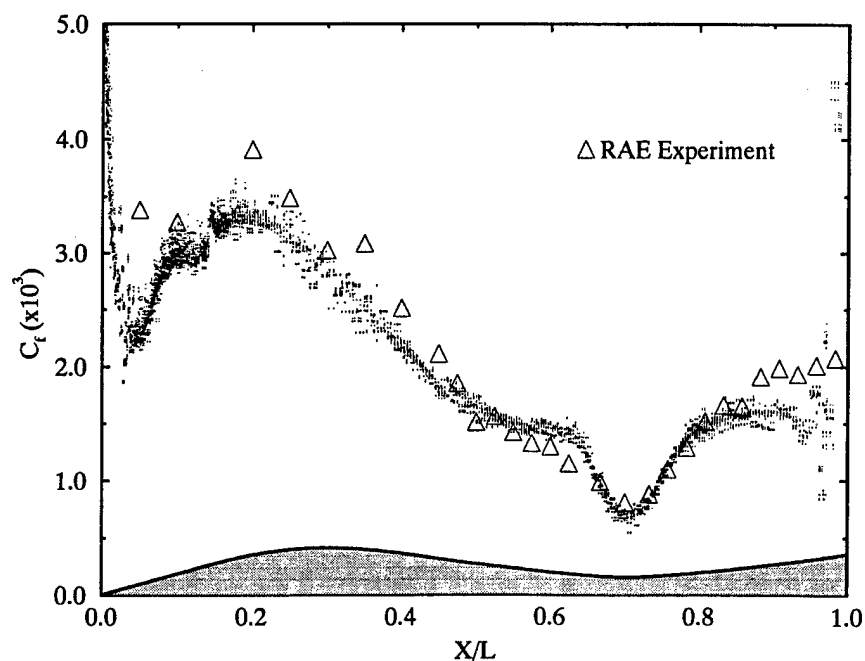

Figure 7.26 Waisted–body local skin friction distribution at Mach = 1.4

The next validation consists of testing the moving mesh boundary conditions for steady viscous flow. This is also a good test of the modifications to the turbulence model. The previous converged Mach = 1.4 viscous solution on the stationary mesh was restarted with a far–field freestream Mach number of 1.3 and a mesh speed of $x_t = -0.1$. This yields a relative Mach number of 1.4 and should converge to yield the same results as the stationary mesh viscous solution. To accelerate the convergence to a steady–state, the x–momentum of the restart file was adjusted to reflect the change in mesh speed.

Figure 7.27 shows pressure coefficient distribution comparisons between the converged Mach = 1.4 solution on the stationary mesh and the converged relative Mach = 1.4 solution with the mesh moving into a Mach = 1.3 freestream. This last solution took 300 iterations at a $CFL = 5.0$ to reduce the RMS of the density residual a few orders of magnitude. The viscous solution pressure coefficient comparisons to each other and to the RAE experimental data are excellent.

Figure 7.28 presents the local skin friction coefficient comparisons between the RAE data, the converged Mach = 1.4 solution, and the moving mesh solution with relative Mach = 1.4. This is a good test of the modification to the Spalart–Allmaras one–equation turbulence model for dynamic motion. The viscous solutions completely agree with each other and both compare very well with the RAE wind tunnel results. It is especially encouraging to see the Spalart–Allmaras turbulence model work so well in the adverse–pressure gradient region between X/L = 0.4 and 0.7.

These viscous solution validations for mesh motion, coupled with the previous inviscid prediction validations, seem to indicate these boundary conditions have been implemented correctly.
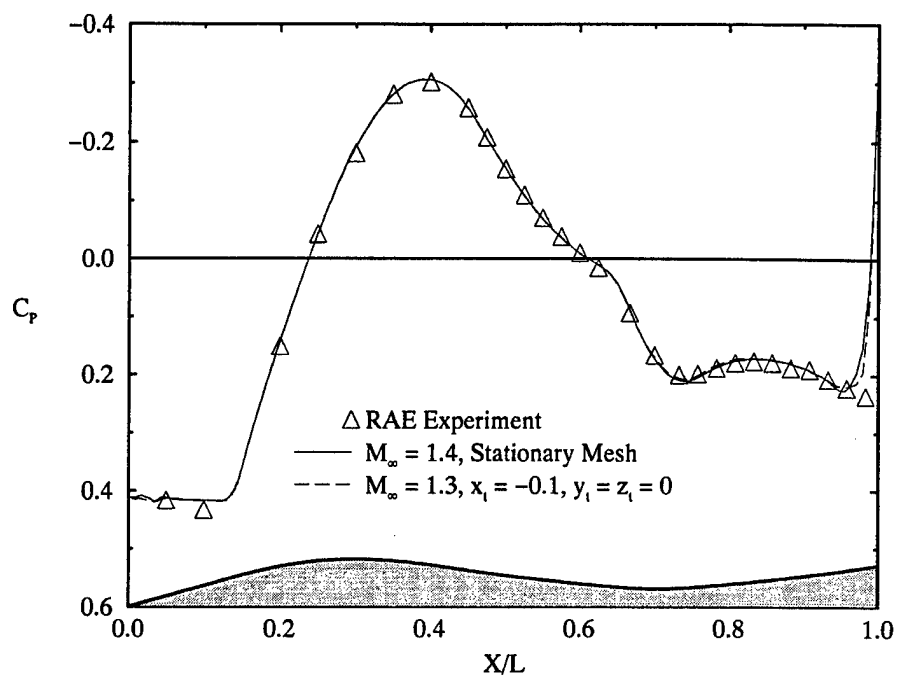
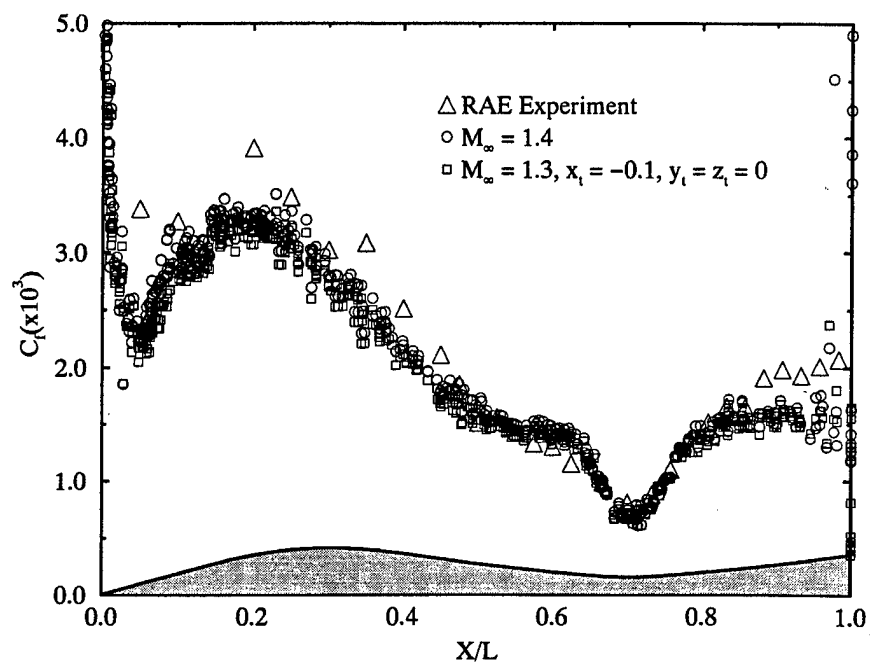Figure 7.27 Pressure coefficient distribution with and without mesh motion

Figure 7.28 Skin friction coefficient with and without mesh motion

7.3 Unsteady Rectangular Wing Solutions

The next logical step in the mesh motion validation is unsteady applications. For the time–accurate unsteady Euler computations the rigid RAE rectangular wing was oscillated about its midpoint periodically with angle of attack according to

$$a(t) = -a_{max}sin(M_\infty kt) \tag{7.1}$$

where the amplitude is $a_{max} = 1°$; the reduced frequency is $k = \omega c/U_\infty$, where ω is the circular frequency (rad./s), c the wing chord length, and U_∞ is the freestream velocity; and t is a nondimensional time based on the wing chord and a_∞, the freestream speed of sound. For these computations $M_\infty = 0.8$ and $k = 0.268$. The steady Euler solution at $M_\infty = 0.8$ and AOA = 0 degrees is used as an initial condition at time zero. Then the numerical mesh is rigidly rotated according to Equation (7.1). The first Fourier components of the real, ReCp, and imaginary, ImCp, pressure coefficients, normalized by a_{max}, are used to compare the numerical and experimental results. These components can also be thought of as in–phase and 90 degrees out–of–phase with the unsteady angle of attack given by Equation (7.1).

A complete oscillation, or cycle, of the RAE rectangular wing at 720 steps/cycle had a time–step of 0.04. Based on a minimum time–step analysis of the steady–state Euler solution, this was an effective maximum *CFL* of 112 for the refined mesh and 43 for the 0.02c mesh. The Euler results to be presented are second–order time–accurate, using Roe's FDS, freezing the analytic Jacobians, three Newton iterations, Gram–Schmidt for linear reconstruction for higher–order spatial accuracy, and the Barth–Jesperson limiter was not used. Additional Euler results using first–order time–accuracy, numerical Ja-

cobians, and Green's theorem for linear reconstruction for higher–order spatial accuracy were virtually identical and will not be presented.

Figure 7.29 presents three cycles for the Euler solutions for the imaginary pressure coefficients at the 50% semispan location. By the second cycle the solution was periodic. All the Euler solutions were run three cycles to ensure periodicity. Figures 7.30(a) through 7.30(f) show comparisons between the refined and the 0.02c mesh and real and imaginary pressure coefficients for the unsteady Euler solutions and experimental data for 50, 77 and 94% semispan, respectively. In these figures the numerical results are plotted for the top and bottom of the wing surface to show the symmetry of the solution. The experimental data was from the upper surface and is indicated with an upright triangle symbol. This data was reflected for comparison to results on the bottom surface and is indicated with an inverted triangle.
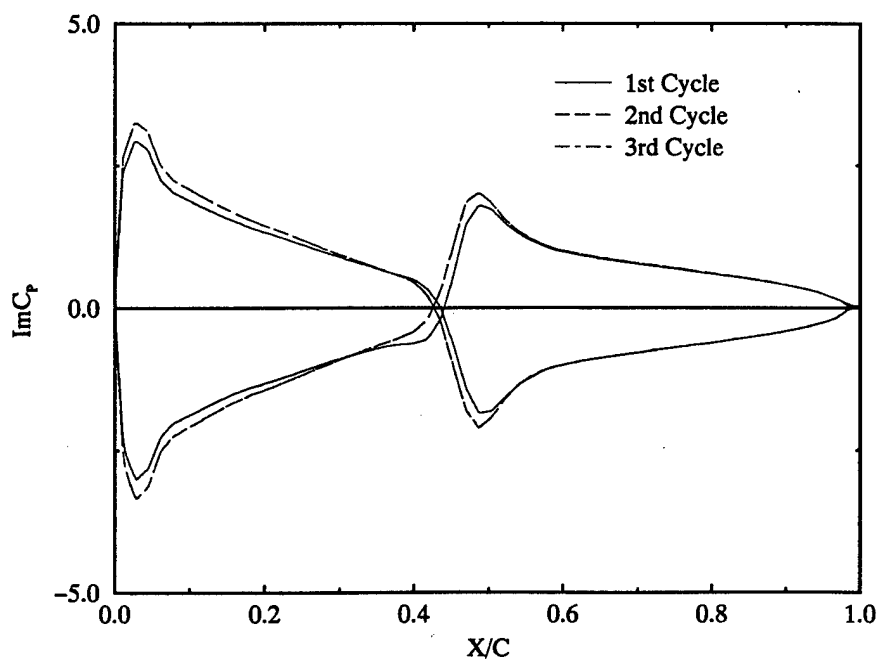
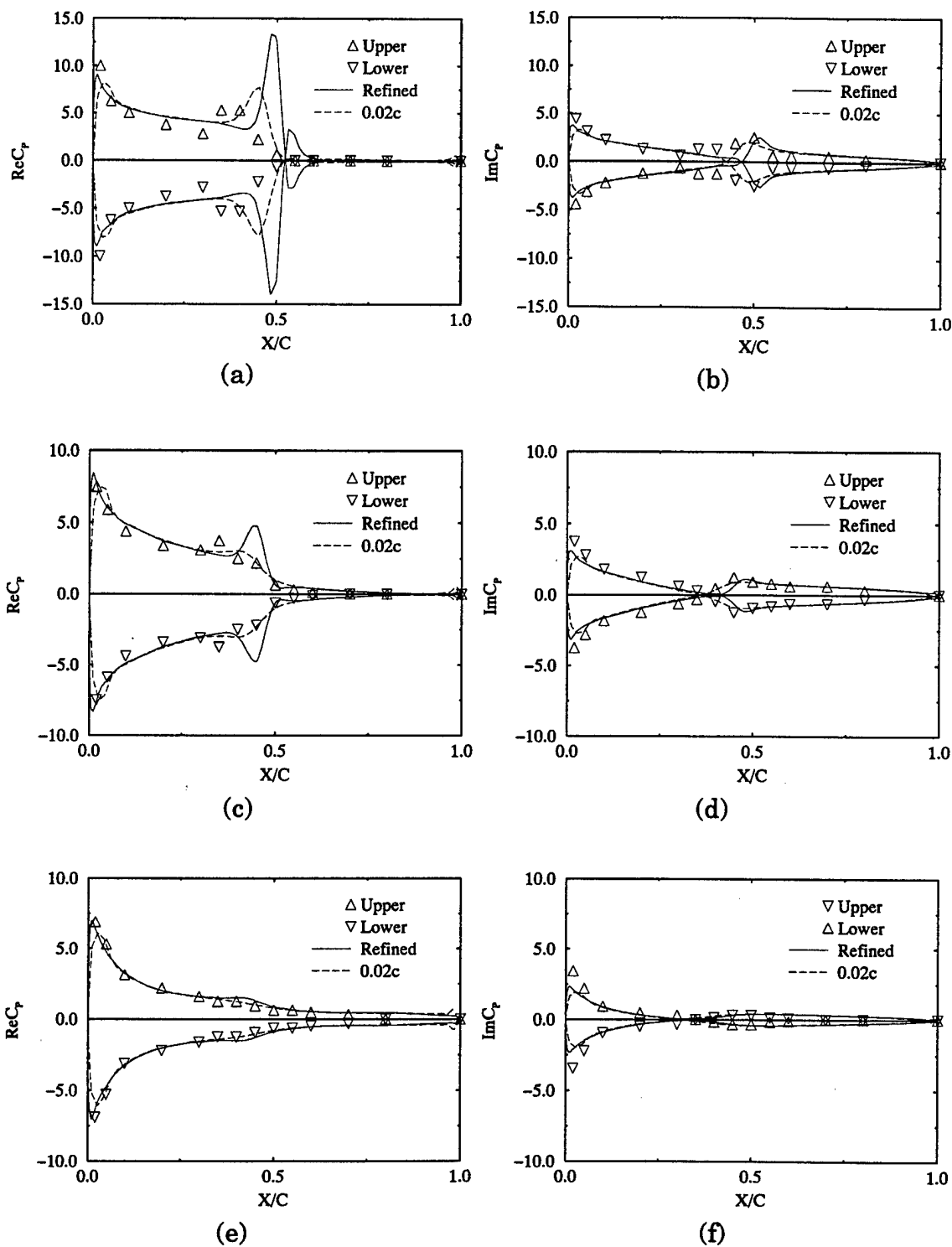Figure 7.29 Periodicity of oscillating rectangular wing Euler solutions

Figure 7.30 Rectangular wing real and imaginary Fourier component pressure coefficient comparisons with unsteady Euler solutions and experiment at semispans: a) and b) 50%; c) and d) 77%; e) and f) 94%

For the 50% semispan location, Figures 7.30(a) and 7.30(b), the refined mesh result compares well to the experimental data in the leading edge region but is slightly aft of the data in the shock excursion region. The 0.02c mesh matches the experimental data better in the shock excursion region. There may be a few reasons for this behavior. It was noted in the RAE report [67] that the experimental data at a semispan location of 17% had been affected by the sidewall boundary layer and could not be relied upon. There may also be an affect at the 50% semispan location as well. Also, in the steady–state solution for the refined mesh in Figure 7.17, the predicted shock was much sharper with a greater peak and slightly aft of the 0.02c mesh solution. Furthermore, it appears the shock moves outside the uniform region centered about the mid–chord location of the refined mesh. This excursion in and out of cells of differing aspect ratios may also adversely influence the time–dependent Fourier coefficients. A uniform mesh of 0.01c was constructed to evaluate this and a discussion of those results will be presented shortly.

For the 77 and 94% semispan locations, Figures 7.30(c) through 7.30(f), the solution trends are similar to the previous results. The 0.02c mesh matches the experimental data better than the refined mesh. In all cases the refined mesh resolves the leading edge unsteady data better than the 0.02c mesh. However, for the region about the mid–chord, which is supercritical for the 50 and 77% semispan locations and slightly subcritical for the 94% semispan location the uniform spacing 0.02c mesh appears to match the experimental data better. Since the experiment was fully turbulent and these are Euler solutions it may very well be that the dissipative nature of the coarse uniform mesh models the true viscous flow field better than the refined mesh.
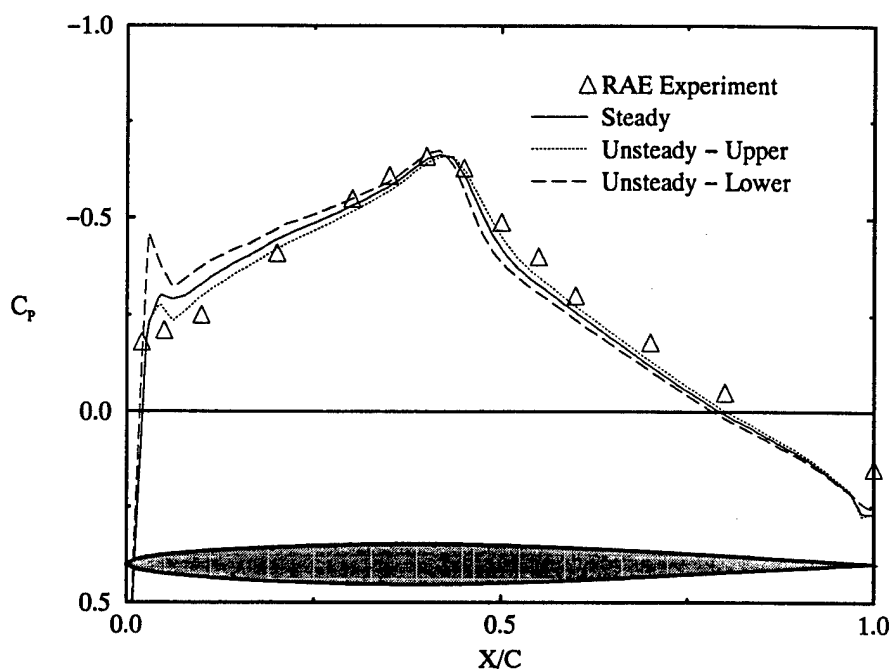
Figure 7.31 Steady and unsteady pressures for oscillating wing

Figure 7.31 presents a comparison of the Euler pressure coefficient distributions at $M_\infty = 0.8$ and $a = 0°$ along the upper wing surface for the steady solution and the upper and lower surfaces for the unsteady, oscillating solution. The unsteady solution is plotted at the instant it completes the third cycle and passes back through $a = 0°$. The oscillation starts with a pitch down to $a = -1°$, pitches back up to $a = 1°$, and then pitches back down through $a = 0°$. The unsteady Euler results at this instant captures the shock motion during the oscillation. Notice the unsteady shock is slightly aft on the upper surface and slightly forward on the bottom surface of the steady solution shock location.

As mentioned, further investigations with a finer uniform mesh were conducted. The uniform mesh spacing was set at 0.01c which resulted in an overall 3–D unstructured mesh of 212,895 nodes with 1,111,104 tetrahedrons.

Figure 7.32 presents steady Euler solutions at Mach = 0.8 and $a = 0°$ at the 50% semipsan location for this finer uniform mesh compared to the selectively refined mesh. The finer 0.01c mesh Euler solution produces a sharper shock than the refined mesh at the 50% semispan. The results at the other semispan locations were close to those for the refined mesh. An unsteady Euler solution produced results that were virtually the same as those for the refined mesh and will not be presented.

These Euler solutions produce sharper shocks at a slightly downstream position compared to the RAE experimental data As expected, this affects the unsteady results and comparisons to the unsteady experimental data. However, the overall trends are correct and the inviscid boundary conditions are considered valid.
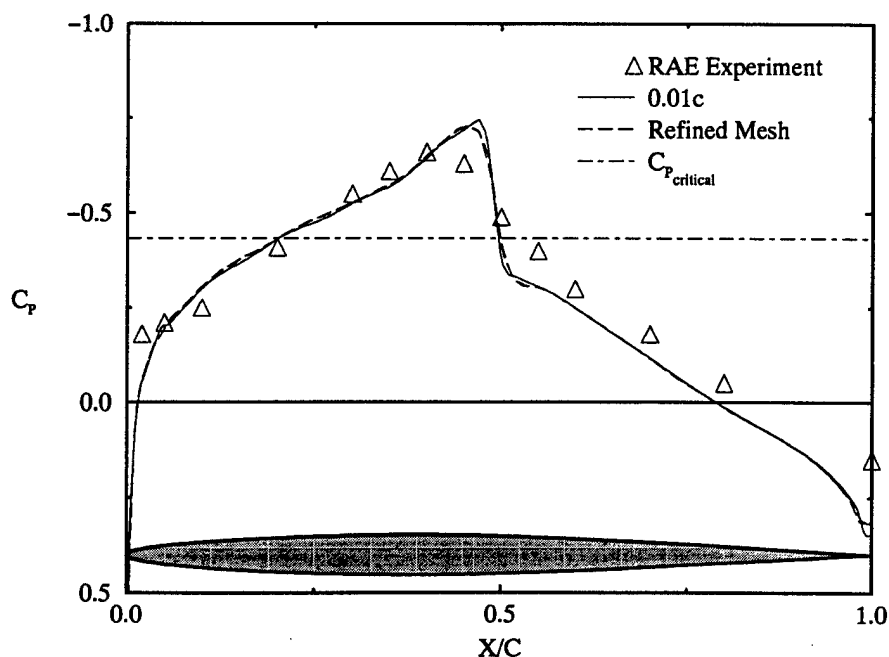

Figure 7.32 Pressure coefficient, refined and 0.01c meshes, semispan = 50%

7.4 Missile Trajectory Simulation

The simulation of a missile trajectory encompasses the coupling of the 3–D unstructured flow solver for the equations of fluid motion with the 6DOF model which solves the rigid body equations of motion. Additionally, the simulations presented in this study incorporate a rocket motor exit boundary condition. Most trajectory simulations start from a steady, dynamically stable condition. Then, various control–surface or motor inputs, such as thrust–vectoring, are added to assess the dynamic reaction of the vehicle's trajectory. For the 3–D unstructured CFD method the first step was to obtain a steady Euler solution, use this as an initial state for the 6DOF, then run the 6DOF coupled with the flow solver until a steady dynamic state is obtained or, if starting from a steady dynamic state, try various inputs to assess the missile's flight reaction. The following sections present the results from two such trajectory simulations – a steady roll and pitch and yaw thrust–vectoring. These trajectory results will follow the discussion of the missile model and motor exit condition in the next section.

7.4.1 Missile and Solid Rocket Motor Models

The missile and solid rocket motor models were built from pictures and data in the open literature, as will be referenced, and are not to be construed as anything other than generic and unclassified. The missile is modeled after the AIM–9X (Air Intercept Missile) Sidewinder which is a short range heat seeking missile with enhanced maneuverability through thrust–vector control [75]. The geometry consists of a spherical nose cap, tapering conically into the cylindrical body, with four delta fins fore and four clipped delta fins aft. The "box size" of the missile is such that it can fit in a rectangular box around 10 feet long with sides less than a foot in length to be carried internally in advanced fighters such as the USAF F–22 and the Joint Strike Fighter.
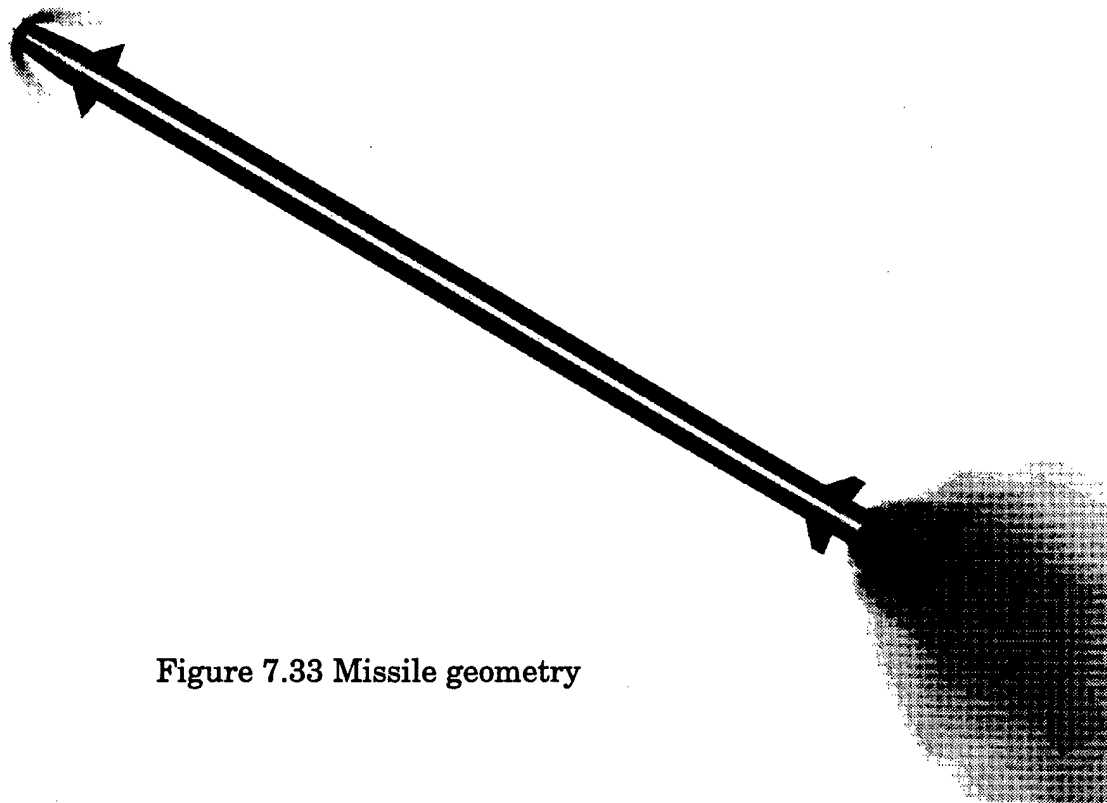
Figure 7.33 Missile geometry

Figure 7.33 shows the missile geometry with exhaust plume at a super-sonic flight condition. The solid rocket motor nozzle exit plane conditions for a typical air–to–air missile exhaust are Mach numbers from 2 to 4, pressures from 1 to 3 atmospheres, and temperatures from 800 to 1,200 degrees Kelvin for standard propellants or 1,500 to 2,500 degrees Kelvin for more energetic propellants [76]. To separate the exhaust plume effects from the pressure distribution along the body the flight conditions for the trajectory simulations were supersonic at Mach 3. If a viscous simulation were run there may be large exhaust plume effects on the missile afterbody if the initial expansion causes the boundary layer to separate. To isolate the evaluation of the coupling of the flow solver to the 6DOF model, and to save computational resources and time, the trajectory simulations were run inviscidly.

Figure 7.34 shows pressure contours in a plane in the fore delta fin region for the missile at Mach 3.0 and zero degrees AOA. Notice the bow shock and a secondary shock off the nose–/body juncture. From the stagnation pressure at the nose there is a rapid expansion along the spherical cap which then starts compressing along the conical portion of the nose. At the juncture of the nose conical portion and the body the flow passes through this secondary shock with a step increase in pressure. After the juncture the flow expands and then rapidly accelerates between the delta fins causing greater expansion. A tertiary shock between the fins compresses the flow once again. After that, the flow gradually expands to the freestream pressure along the missile body until another rapid acceleration between the rear clipped delta fins.
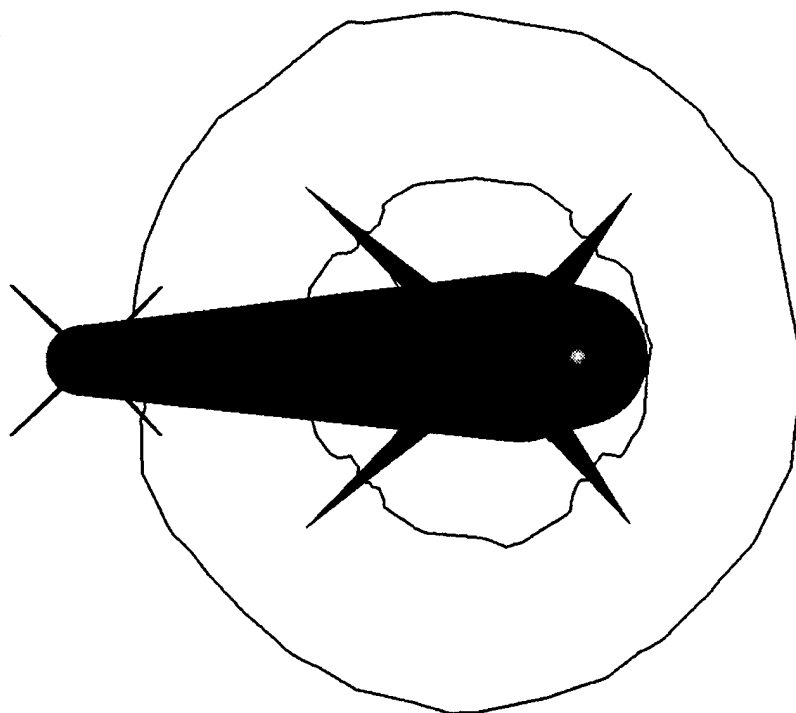

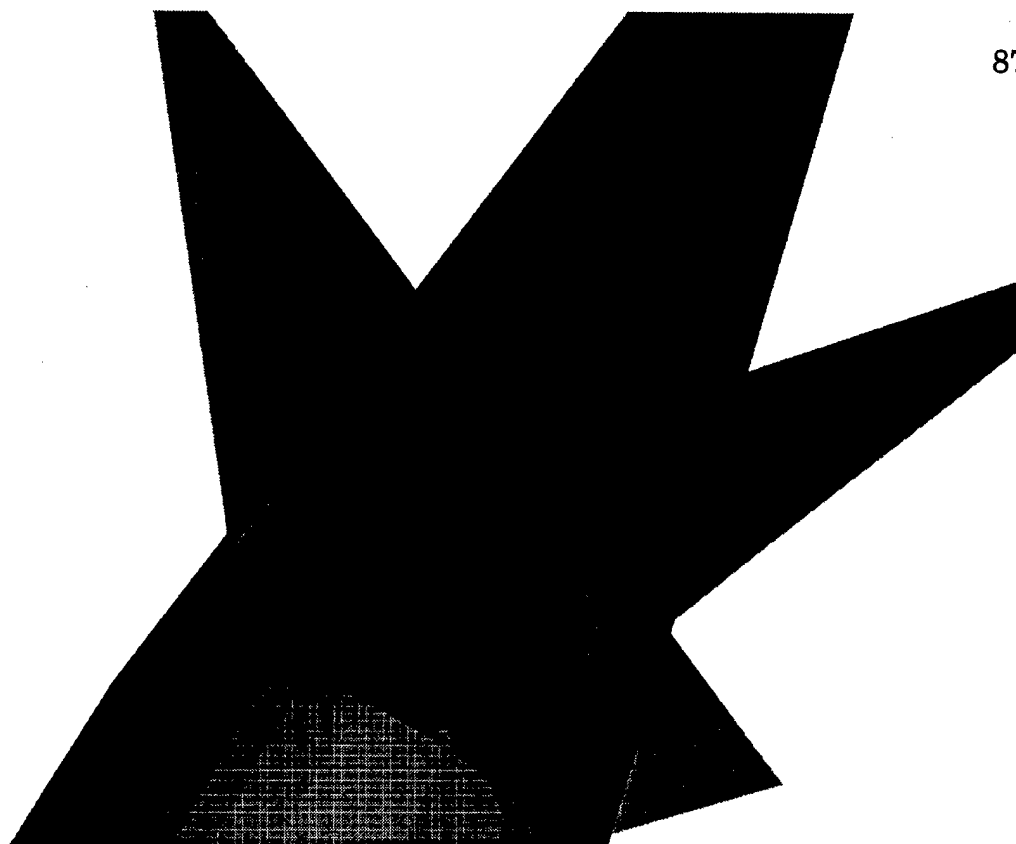Figure 7.34 Nose view of missile with bow and secondary shock

Figure 7.35 Missile body surface pressure contours between fore delta fins

Figure 7.35 is a contour plot of the pressure distribution near the conical nose and cylindrical body juncture and between the fore delta fins. The small compression shocks off the leading edge of the fins coalesce at the centerline. The pressure distributions on the fins seem to indicate a secondary shock from about a quarter of the root chord of the base of the fin to about 75% semispan of the fin trailing edge. The qualitative features of this complex 3–D region was captured quite nicely by the Euler solver.

In the next couple sections the simulation trajectories will be discussed. The initial solutions were run inviscidly with Roe's FDS, the Gram–Schmidt least squares procedure for higher–order spatial accuracy, the Barth–Jesperson limiter and local time–stepping. The first 500 iterations were first–order spatial accuracy and then higher–order was switched on for 1,500 iterations for a total of 2,000 iterations. A *CFL* = 1.0 was required for the initial transients due to the motor condition and was used throughout. It could have been

bumped up sooner, but an initial solution took a little over 5 hours at around 5.57e–04 seconds per node per iteration on an SGI Onyx 2 R10000 which runs at 250MHz and has 4MB secondary cache. By comparison, the trajectory simulations ran at around 1.8e–03 seconds per node per iteration with 3 Newton iterates, first–order temporal accuracy, $CFL = 2.5$, and time–step sorting and truncating.

The time–step sorting and truncating during a trajectory simulation was used to allow a larger global time–step than a straight minimum time–step would allow. For the missile trajectory simulations the motor exit condition presented the severest case for the flow solver and dictated a very small time–step. Figure 7.36 is a histogram plot comparing the percentage of the total nodes in the computational domain to the logarithm of the predicted time–step from inviscid stability analysis as given in Chapter V. The number of nodes with the restrictive time–step is less than a few percent of the total nodes in the computational domain. Figure 7.37 is a contour plot of the time–step distribution along the missile's motor exit plane where the lighter grays-cales indicate the smaller time–steps. By allowing these relatively few node–centered cells to advance temporally with local time–stepping and using a global time–step which is a minimum for the rest of the cells the solution can be run at a larger time–step without adversely affecting the global time accu-racy. The cut–off percentage was kept at 1% for the trajectory simulations to be presented. The sorting was done at each time–step to ensure the simula-tion did not step over any salient temporal features should they arise. With a CFL of 2.5 and using the effective CFL analogy discussed earlier effective CFL numbers up to order 10 were used. A higher cut–off percentage allows a high-er CFL, but the effect on the overall temporal accuracy may be compromised.
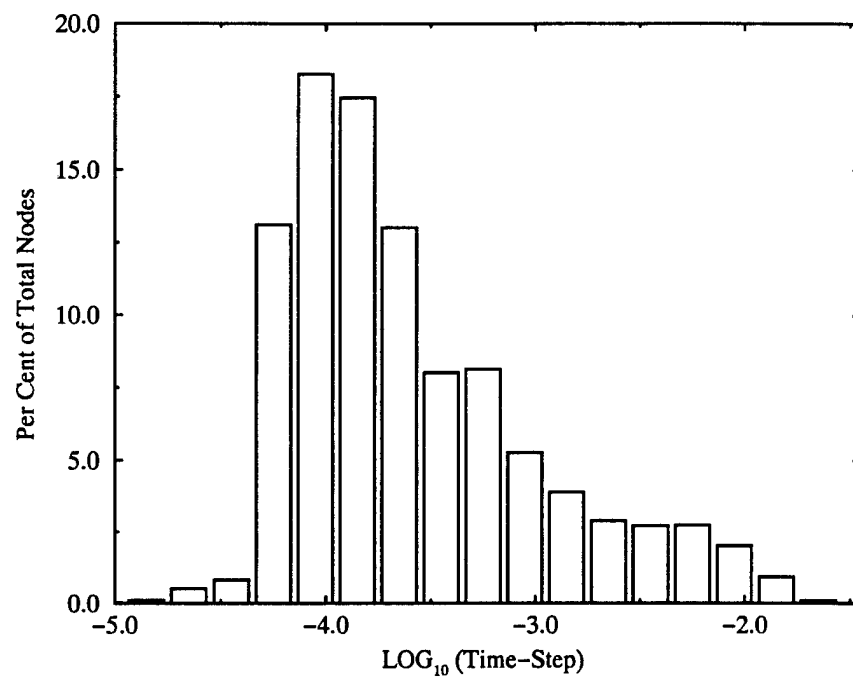
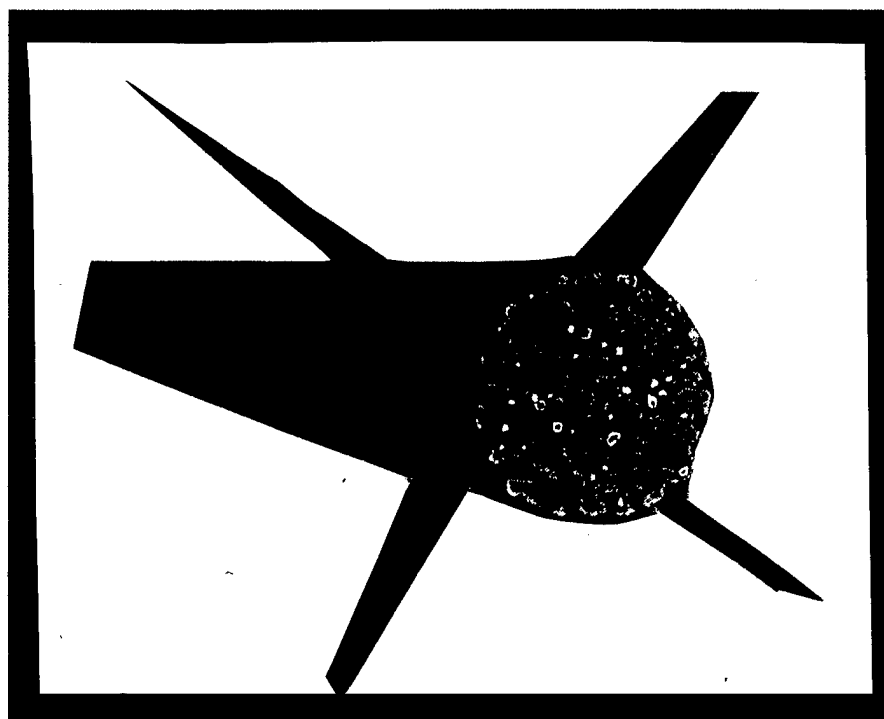Figure 7.36 Histogram of percentage of total nodes to LOG_{10} (time–step)

Figure 7.37 Contour plot of time–step distribution on missile exit plane

7.4.2 Roll Trajectory Simulation

The first missile simulation begins with the missile in flight with a roll angle close to 20° off the dynamically stable "X" cruciform position as shown in Figure 7.38(a). The first step was to obtain a steady–state Euler solution as an initial condition. The simulation is set at an altitude of 17 km with the missile flying at Mach 3. The solid rocket motor exhaust exit conditions were 1.3 atm, 1,070 °K, and an exit Mach number of 3.0. This gave an exit to ambient pressure ratio of 15.5 and allowed for isentropic expansion of the exhaust stream to a maximum Mach number of 5. The initial position of the CG was at the center of the missile (nose–to–exhaust) at inertial coordinates of $x = -2.5$, $y = -0.47$, and $z = 1.69$. The angular position and angular velocity state variables were initialized to zero. The simulation was run for 10,000 iterations to a non–dimensional time just under 1.0. This corresponds to an actual time of about 0.01 seconds, based on the non–dimensionalizing quantities of missile nose–to–exit plane length, $L = 3.1$ m, and the freestream speed of sound for this altitude, $a_\infty = 295.1$ m/s. A longer simulation could be run, but at these speeds the missile moves 2.5 body lengths and the salient steady roll trajectory features are evident.
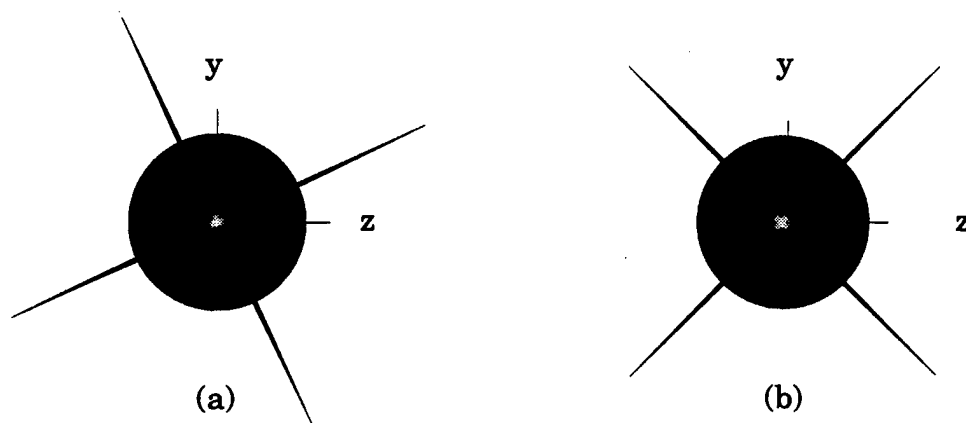

Figure 7.38 Missile roll trajectory simulation initial (a) and final (b) position.
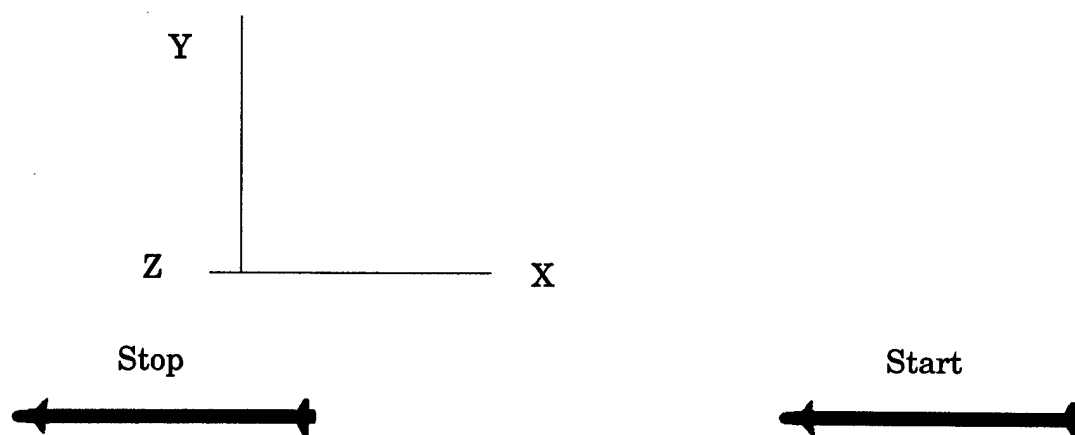
Figure 7.39 Missile roll trajectory start and stop positions

Figure 7.39 graphically displays the start and stop position of the missile during the trajectory simulation. The motor exhaust is not displayed. Figure 7.40 shows the Euler angle positions in non–dimensional units (radians) during the trajectory. Notice the positive roll angle which indicates the missile has a right–"wing" down roll as would be expected for this initial missile orientation to move to a dynamically stable flight position. The positive yaw, although difficult to see compared to the roll angle, indicates the nose is moving to the right. Figure 7.41 plots the rate of the Euler angle changes. As expected the roll rate is much greater than the pitch or yaw rates. The pitch angle and rate changes during the simulation were also of interest. At these flight conditions and at this altitude there was not enough lift generated and the missile initially pitched down and yawed to the right. As the missile rolls to the "X" cruciform position the fore delta fins start to generate lift in the "up" direction of the missile body reference frame, the pitch rate goes positive and the missile pitches up into a positive angle of attack. This is graphically depicted in Figures 7.42 and 7.43 which plots the pitch angle and pitch rate time history during the roll trajectory simulation.
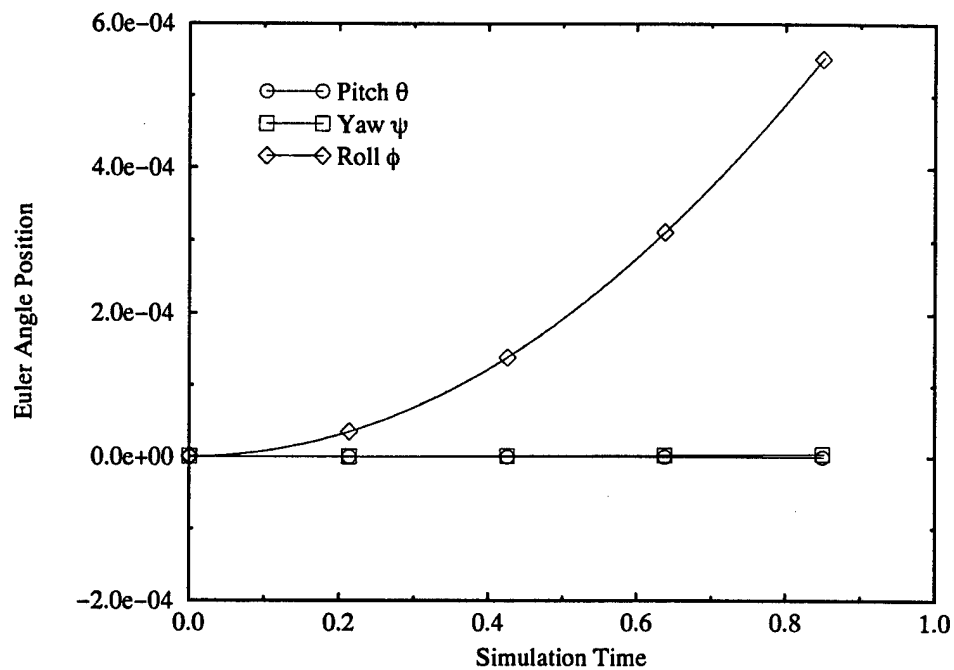
Figure 7.40 Euler angle positions during roll trajectory simulation
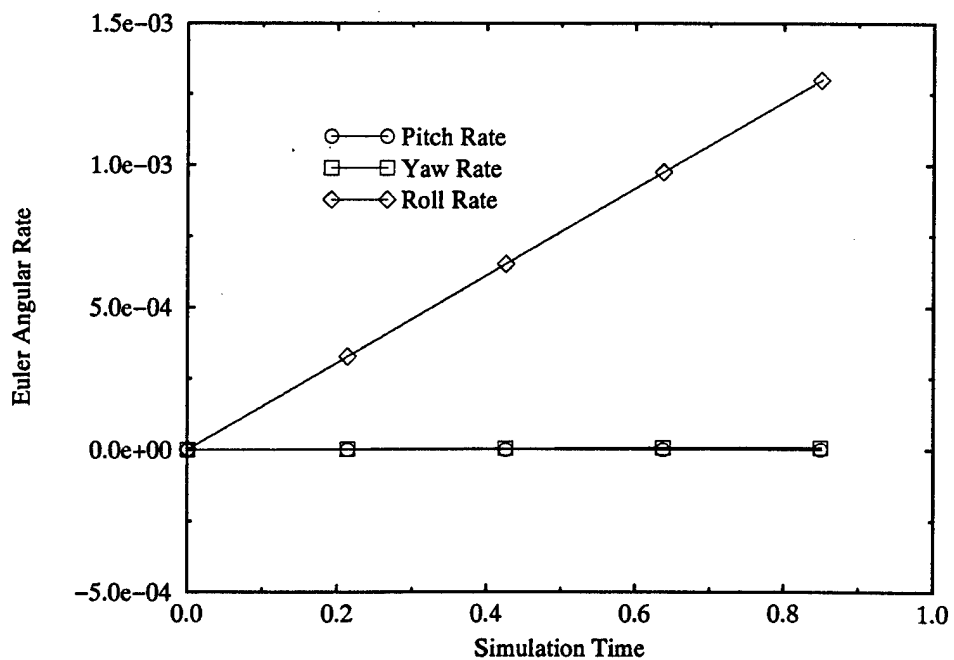

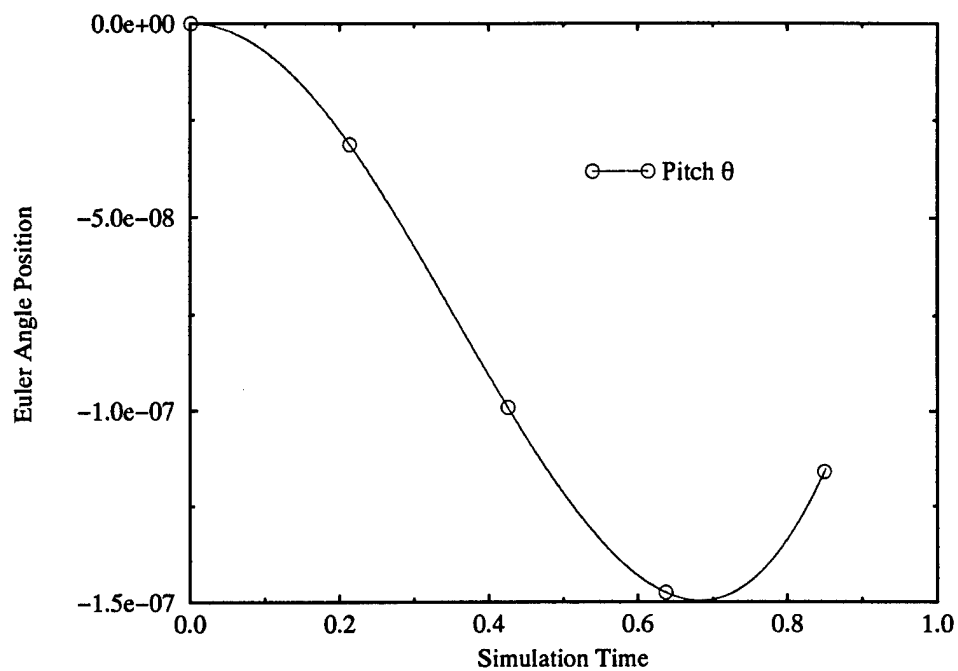Figure 7.41 Euler angle rates during roll trajectory simulation

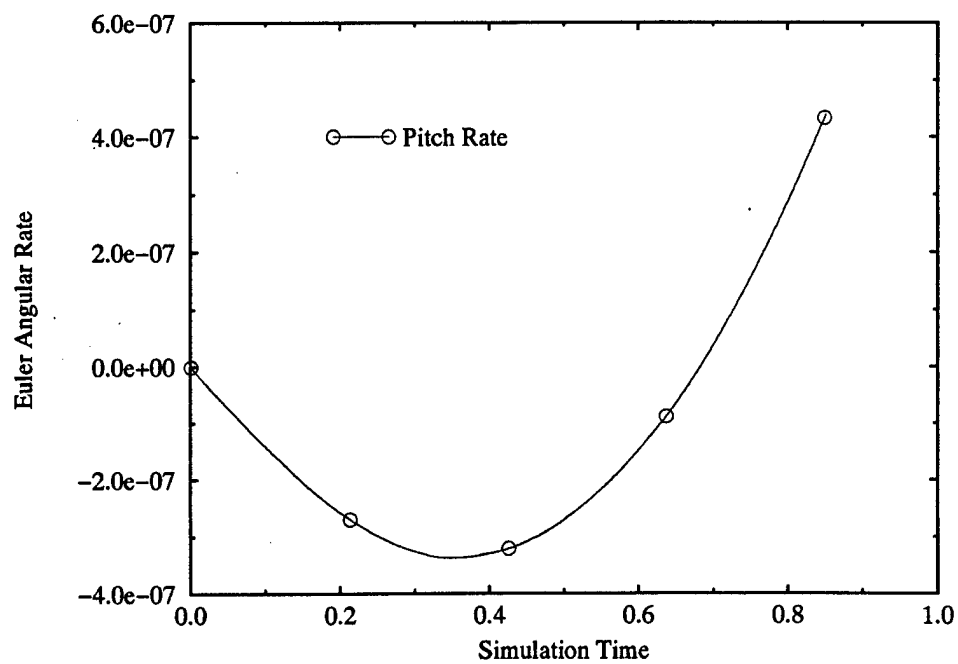Figure 7.42 Pitch angle during roll trajectory simulation

Figure 7.43 Pitch angle rate during roll trajectory simulation

7.4.3 Thrust–Vectoring Trajectory

The second missile simulation begins with the missile in flight in the dynamically stable "X" cruciform position as shown in Figure 7.38(b). Once again, the first step was to obtain a steady–state Euler solution as an initial condition. This simulation is set at an altitude of 7 km with the missile flying at Mach 3. The solid rocket motor exit conditions were 6.7 atm, 1,430 °K, and an exit Mach number of 3.0. This gave an exit to ambient pressure ratio of 16.5 and allowed for isentropic expansion of the exhaust stream to a maximum Mach number of 5. The initial CG position was at the center of the missile (nose–to–exhaust) at non–dimensional inertial coordinates all at the origin – $x = y = z = 0$. The angular position and velocity state variables were initialized at zero. The simulation was run for 10,000 iterations to a non–dimensional time just under 1.0. This corresponds to an actual time of about 0.01 seconds, based on the non–dimensionalizing quantities of missile nose–to–exit plane length, $L = 3.1$m, and the freestream speed of sound at this altitude, $a_\infty = 312.3$m/s. A longer simulation could be run, but at these speeds the missile moves 2.5 body lengths and the salient steady roll trajectory features are evident.

As explained in Chapter VI, the solid rocket motor exit condition prescribes the Mach number, density and pressure at the exit plane surface. The thrust–vector control is handled by rotating the subsequent velocity and force vectors, which are normal to the surface, to the prescribed yaw or pitch angle, utilizing the following rotation matrix

$$B_{TV} = \begin{bmatrix} \cos\theta\cos\psi & -\sin\theta & \cos\theta\sin\psi \\ \sin\theta\cos\psi & \cos\theta & \sin\theta\sin\psi \\ -\sin\psi & 0 & \cos\psi \end{bmatrix} \tag{7.2}$$

7.4.3.1 Pitch Thrust–Vector Trajectory

The first trajectory simulation using thrust–vectoring is a comparison between the missile's flight paths at two different altitudes during a preprogrammed series of pitch maneuvers. In a missile flight test program a controlled–vehicle test is usually conducted whereby "step acceleration commands are programmed in the missile to occur at specified intervals during the controlled flight [3]." Hence, this simulation used step thrust–vectoring commands. An actual servo–motor for thrust–vector control vanes would take fractions of a second to rotate into position.

The flight altitudes selected for the simulation are 7 and 17 km – the lower altitude case more representative of where a tactical missile would be launched in air–to–air combat. The 17 km missile used stronger motor conditions than those for the roll trajectory simulation. The solid rocket motor exit conditions were 2.7 atm, 1,250 °K, and an exit Mach number of 3.0. This gave an exit to ambient pressure ratio of 31 and allowed for isentropic expansion of the exhaust stream to a maximum Mach number of 5.7. This resulted in the missile at 17 km having an exit to ambient pressure ratio close to twice the ratio for the 7 km missile – the net results being it had almost twice the thrust of the 7 km missile. This is offset by the fact the 7 km missile is operating at a higher altitude where the density of air is 75% less than the density of air at 7 km. Therefore, the results will show predominantly altitude effects.

The thrust–vector pitch angles were alternated between ± 45° using step thrust–vectoring commands. The simulation was run for 10,000 iterations using a 1% cut–off value for the time–steps which gave an effective *CFL* of 2.8 for both cases. The simulation time comes to just under 0.6 which equates to an actual time of 0.006 seconds.
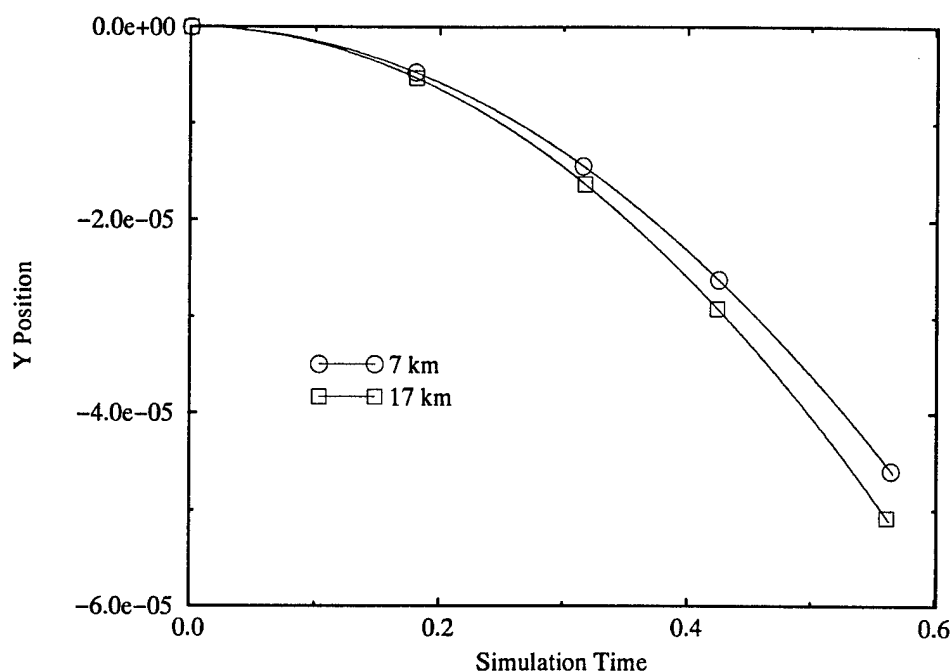
Figure 7.44 Y position during missile pitch thrust–vector maneuvers

Figure 7.44 shows the change in each missile's altitude during the pitch maneuvers. The missile's weight and the low amount of lift generated, even when the thrust–vectoring gives a positive pitch angle, was simply not enough to keep the missile from losing altitude. These results show the altitude effects because the 17 km missile has a greater sink rate than the 7 km missile.

Figures 7.45 to 7.48 depict the pitch schedule via the pitch angles, pitch rates, roll angles and roll rates, respectively, during the maneuver. In Figures 7.45 and 7.46, the step thrust–vector command effects are shown in the slope changes in the pitch angles and pitch rates. In Figures 7.47 and 7.48 both missiles take on a right–wing down roll attitude. The thrust–vectoring affects the roll rate. When the missiles pitch down the roll rate increases. The predominant altitude effects are also evident. The 17 km missile, with the greater thrust, has a flatter pitch and roll angle schedule than the 7 km missile.
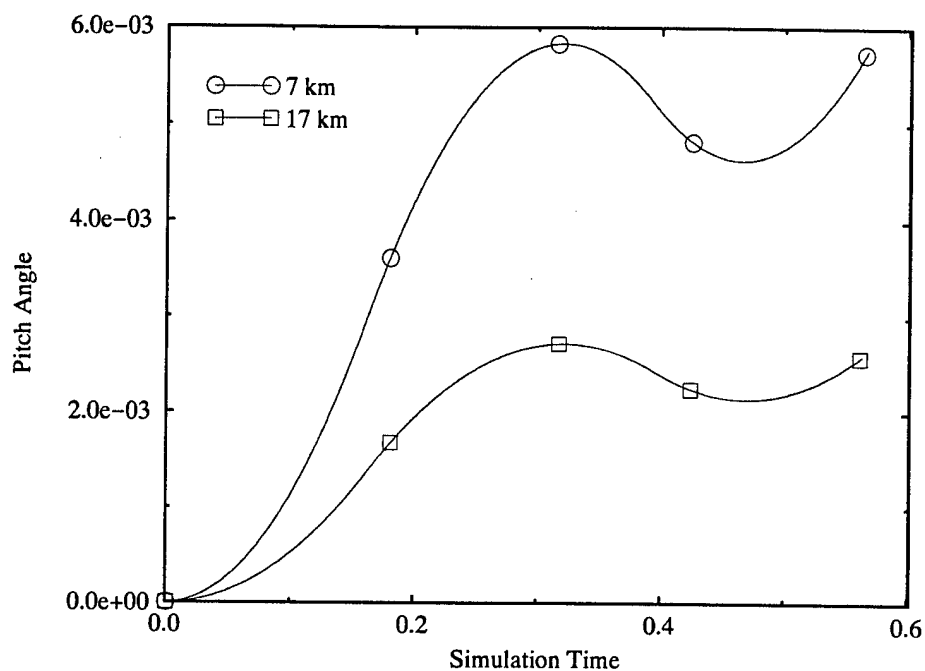
Figure 7.45 Pitch angle during missile pitch thrust–vector maneuvers
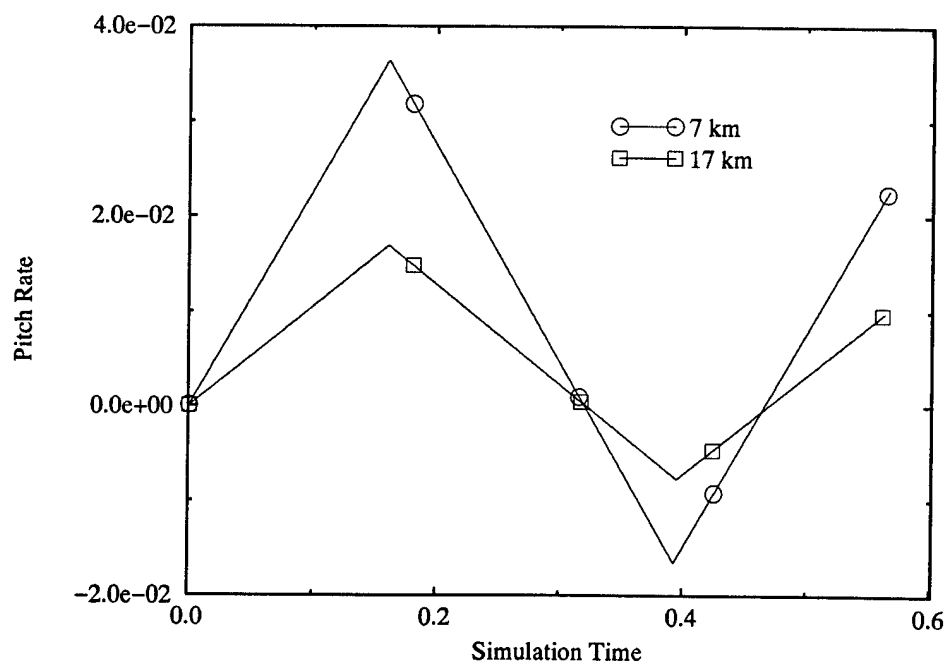

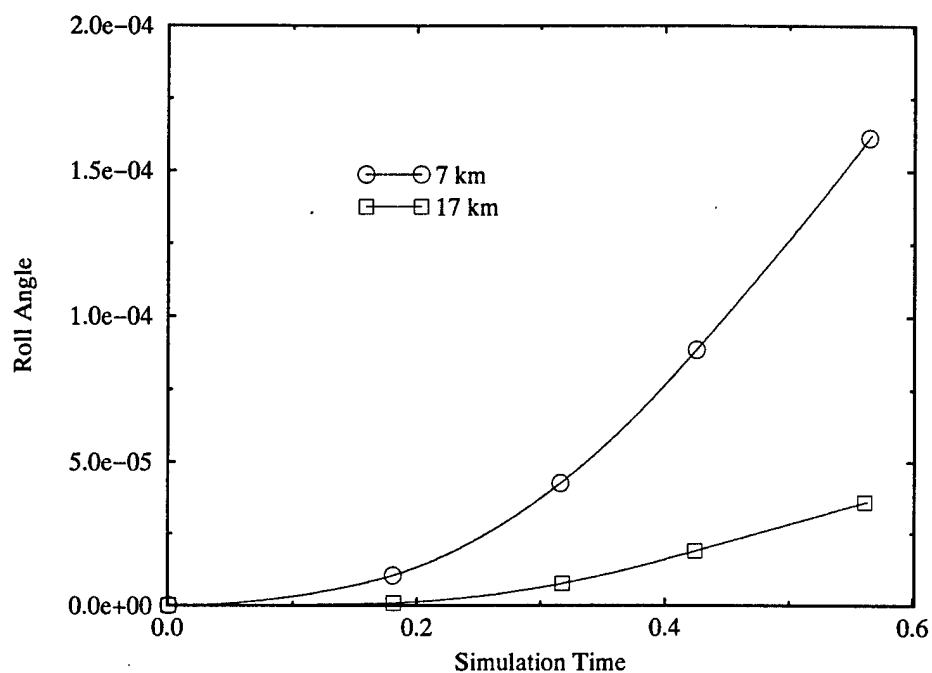Figure 7.46 Pitch angle rate during missile pitch thrust–vector maneuvers

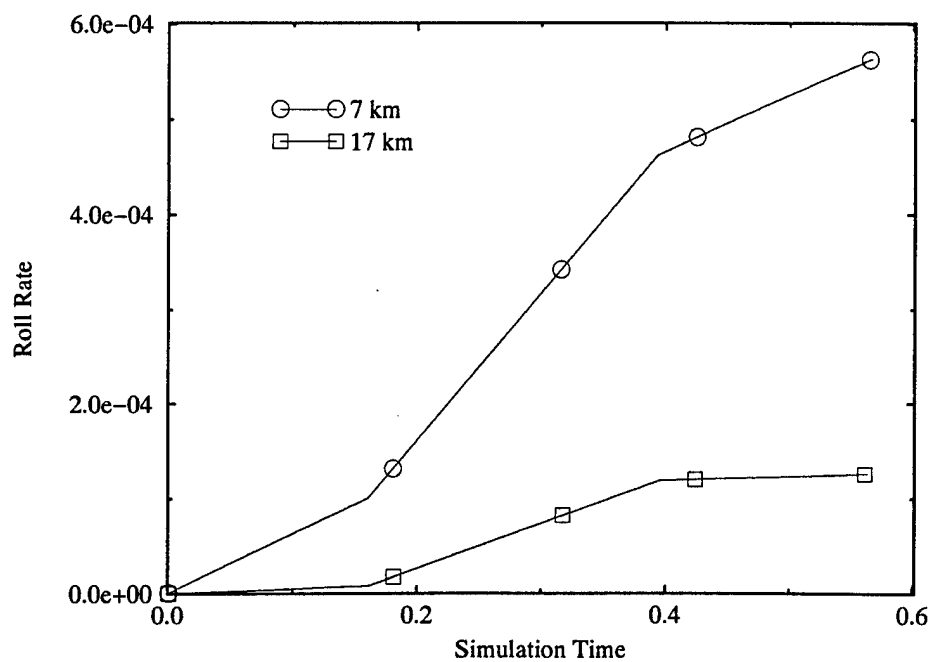Figure 7.47 Roll angle during missile pitch thrust–vector maneuvers

Figure 7.48 Roll angle rate during missile pitch thrust–vector maneuvers

7.4.3.2 Yaw Thrust–Vector Trajectory

The second trajectory simulation using thrust–vectoring is a comparison between the missile's flight paths using different time–step cut–off percentages during a continuous yaw maneuver. This was done to see if the predicted flight–path is adversely affected by allowing a greater percentage of the cells to advance with local time–steppping. This larger cut–off value allows the simulation to run at a larger global time–step. If the Euler solutions for the fluid motion at the larger global time–step do not unduly influence the flight–path results from rigid body motion then it may be possible to run these type of simulations quicker.

The flight altitude selected for the simulation is 7 km with the solid rocket motor exit conditions given in the previous section. The time–step cut–off percentages were 1% and 5% using the technique discussed previously. From the histogram plot in Figure 7.36, a 1% cut–off corresponds to the first two "bars" from the left. A 5% cut–off corresponds to the first four "bars" from the left which includes a significant jump in the minimum time–step. With a *CFL* of 2.5 the 1% and 5% cut–off values yielded global time–steps of 7.9e–05 and 2.25e–04, respectively. This is an increase in global time–step of 2.84. The effective *CFL* numbers, that is, the global time–step divided by the minimum time–step, were 2.85 and 8.13, respectively. The thrust–vector yaw angles were alternated between $\pm$ 45° using step thrust–vectoring commands. The simulation was run for 8,500 iterations using the time–steps indicated for final simulation times of 0.67 and 1.92, respectively.

Figure 7.49 is a plot of the change in the X inertial position during the course of the yaw thrust–vector maneuver. It appears the larger time–step has no affect has no affect on this trajectory and allows the missile to move 6

body lengths. The Y and Z position changes follow the same trend. This would lead one to conclude there may be no harm in allowing the larger global time–step in the Euler solutions of the fluid motion.
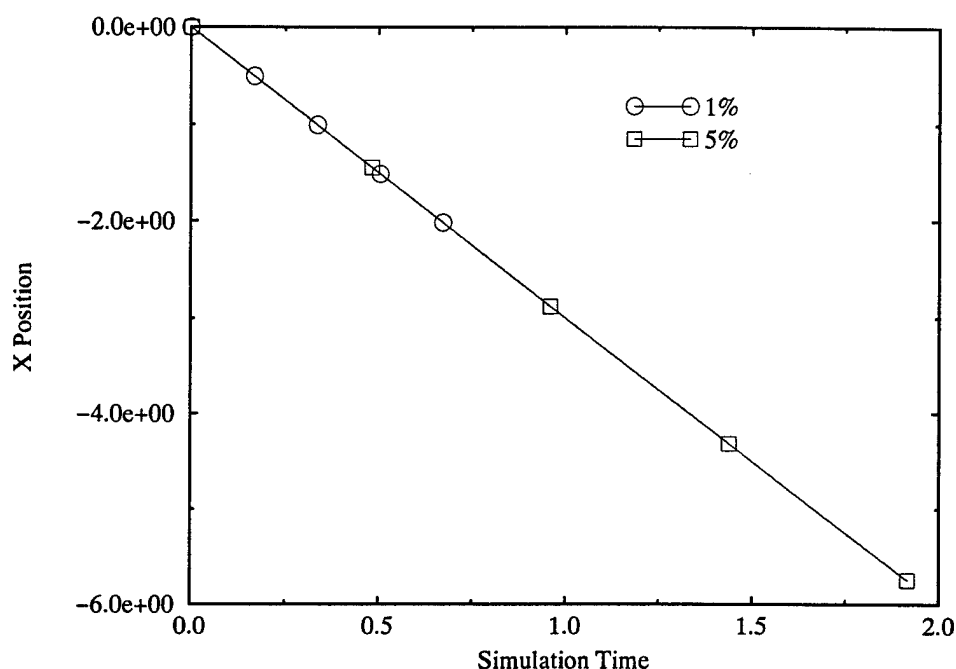

Figure 7.49 X position during missile yaw thrust–vector maneuvers

Figures 7.50 and 7.51 depict the yaw angle and yaw rate, respectively. These trajectory curves also appear to be unaffected by the larger global time–step. Figures 7.52 and 7.53 present the roll angle and roll rate, respectively. These curves are of the same relative magnitude as the yaw angle and yaw rate curves. However, in this case the roll angles take distinctly different paths during the two trajectories. With the smaller 1% cut–off the roll angle and roll rate increase during the yaw thrust–vector maneuver is greater. This means there are fluid motion affects that must be accounted for during the maneuver. The increased global time–step using the larger percentage cut–off would miss this trend in the yaw thrust–vector maneuver.
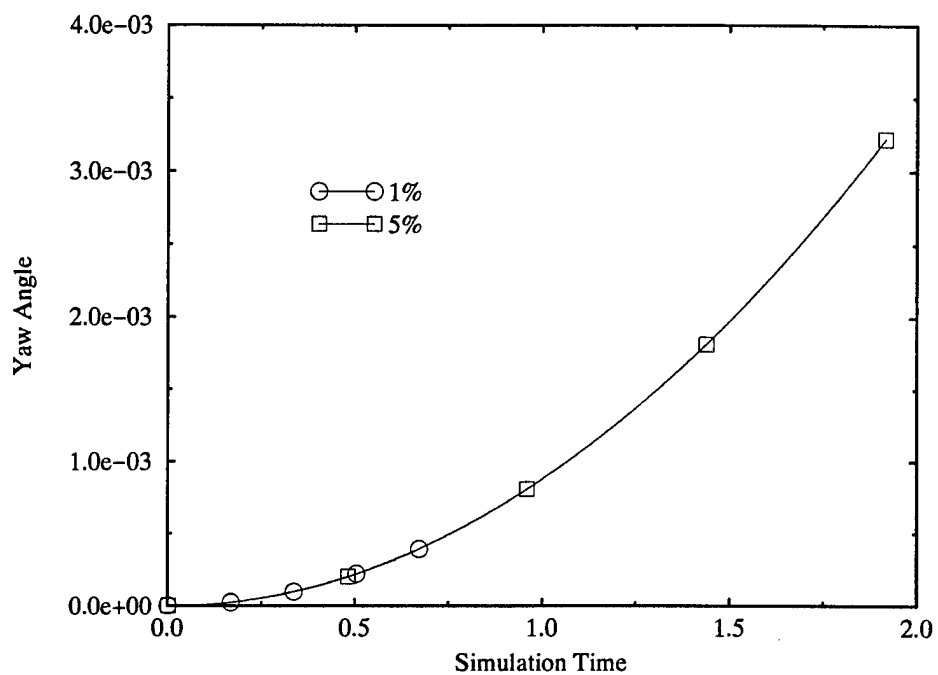
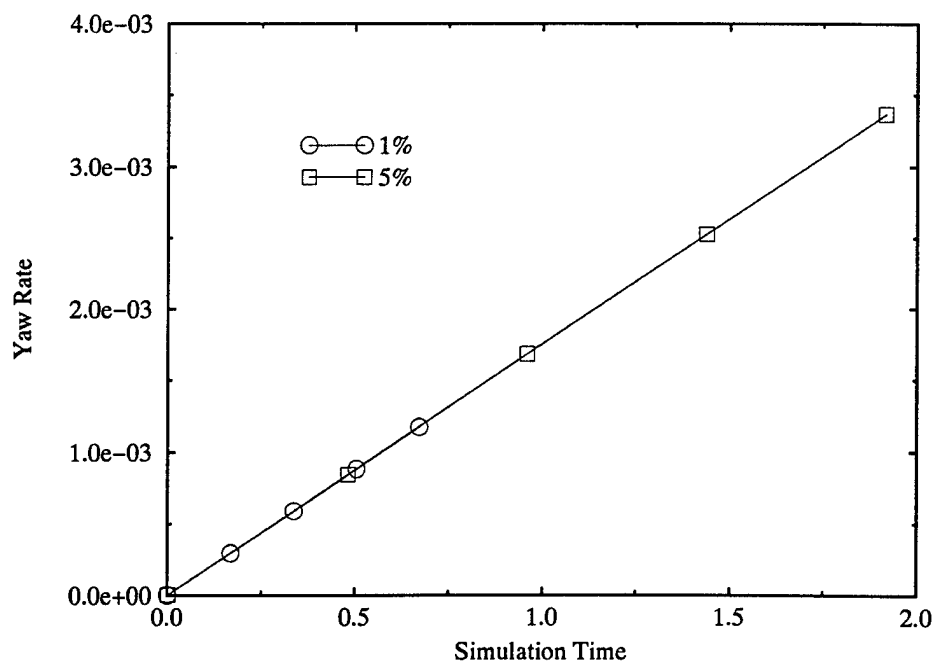Figure 7.50 Yaw angle during missile yaw thrust–vector maneuvers

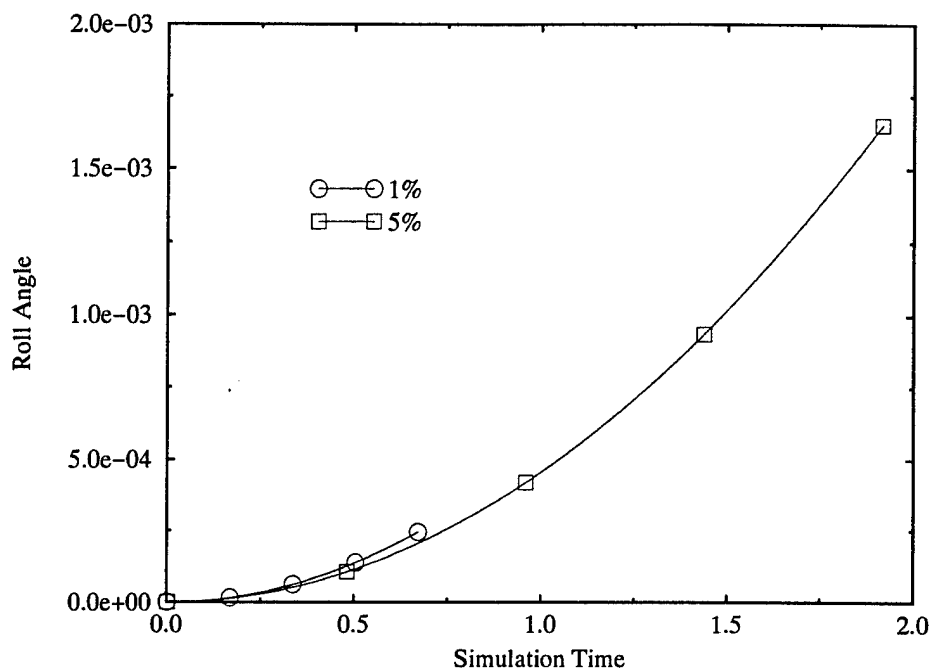Figure 7.51 Yaw angle rate during missile yaw thrust–vector maneuvers

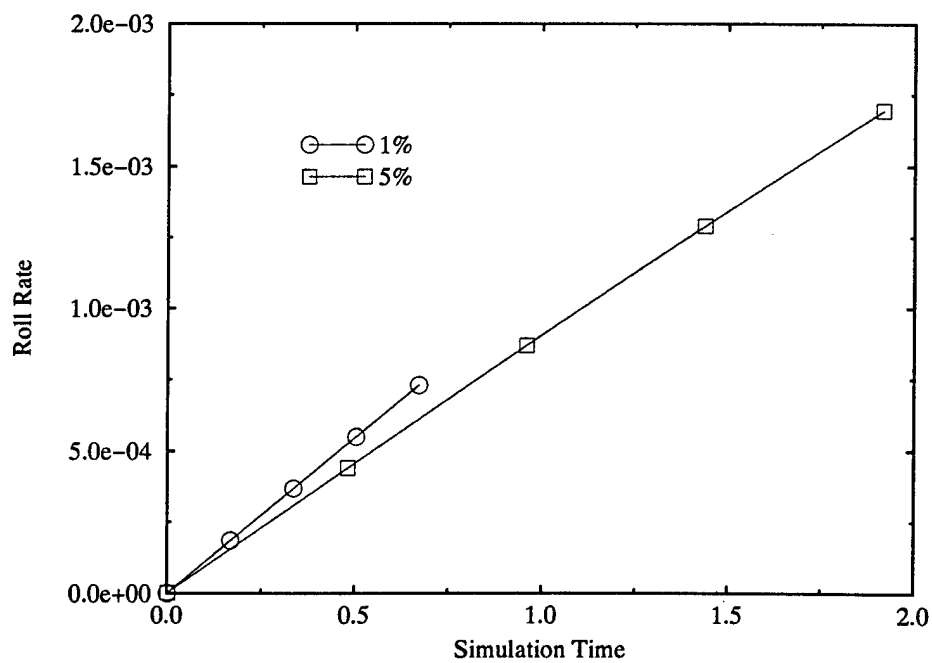Figure 7.52 Roll angle during missile yaw thrust–vector maneuvers

Figure 7.53 Roll angle rate during missile yaw thrust–vector maneuvers

CHAPTER VIII
SUMMARY AND CONCLUSIONS

Development and validation of a 3–D unstructured CFD method for maneuvering vehicles with innovative boundary conditions has been documented in this study. The Euler and Navier–Stokes equations of fluid motion were cast in an Arbitrary Lagrangian–Eulerian frame of reference. Innovative inviscid, viscous, far–field, and rocket motor exit plane boundary conditions were developed and validated for mesh motion. The Spalart–Allmaras turbulence model was also modified to account for mesh motion and validated. Inviscid steady relative motion solutions were validated with experimental pressure coefficient data for two 3–D wings – one delta and one rectangular. Viscous steady relative motion solutions and the turbulence model modifications were validated with experimental pressure and skin friction data for a waisted–body of revolution. Unsteady inviscid solutions using Newton's method for the time–accuracy was validated with experimental data for a 3–D oscillating rectangular wing.

A 6DOF model was developed using Euler angles and the Flat–Earth equations. The 6DOF was explicitly coupled with Euler fluid motion solutions and used to simulate, for the first time, an advanced tactical missile trajectory in free flight and with pitch and yaw thrust–vectoring. Techniques to allow greater global time–steps were implemented. The trajectory results appear very promising for preliminary design and evaluation of advanced missile de-

signs. Application to more complex flight vehicles and maneuvers is definitely a recommended course of action.

Additional recommendations for future work include a complete evaluation of an unsteady viscous solution. The oscillating RAE rectangular wing would be a good starting point. Trajectory validations for the coupled 6DOF and flow solver using experimental and flight test data would be very interesting. Then a missile at high angles of attack, pitch and yaw, would be an impressive virtual simulation of a missile trajectory.

Finally, implementing this technology in a way to allow studies involving multiple bodies in relative motion would be the an ultimate culmination of a 3–D unstructured CFD method for maneuvering vehicles.

REFERENCES

[1] Toro, E. F., *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*, Berlin, Springer–Verlag, 1997.

[2] Ravi, L., "First Look, First Shot, First Kill," *Combat Aircraft: The International Journal of Military Aviation,* Vol. 1, No. 2, July 1997, pp. 76–82.

[3] Eichblatt, E. J., Jr, Editor, *Test and Evaluation of the Tactical Missile*, Progress in Astronautics and Aeronautics, Vol. 119, AIAA, Inc., 1989.

[4] *Proceedings of the First AFOSR Conference on Dynamic Motion CFD,* Rutgers University, New Brunswick, New Jersey, June 1996.

[5] Venkatakrishnan, V., "Perspective on Unstructured Grid Flow Solvers," *AIAA Journal,* Vol. 34, No. 3, March 1996, pp. 533–547.

[6] "Special Course on Unstructured Grid Methods for Advection Dominated Flows," AGARD Report 787, AGARD, 1992.

[7] Mavriplis, D. J., "Unstructured Mesh Generation and Adaptation," 26th Computational Fluid Dynamics Lecture Series, von Karman Institute for Fluid Dynamics, March 1995.

[8] Jameson, A., Baker, T. J., and Weatherill, N. P., "Calculation of Inviscid Transonic Flow over a Complete Aircraft," AIAA Paper 86–0103, January 1986.

[9] Van Leer, B., "Towards the Ultimate Conservative Difference Scheme V A Second Order Sequel to Godunov's Methods," *Journal of Computational Physics,* Vol. 32, 1979, pp 101–136.

[10] Roe, P., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics,* Vol. 43, 1981, pp. 357–362, and reprinted in Vol. 135, 1997, pp. 250–255.

[11] Anderson, W. K., "A Grid Generation and Flow Solution Method for the Euler Equations on Unstructured Grids," *Journal of Computational Physics,* Vol. 110, 1994, pp. 23–38.

[12] Barth, T. J., and Jesperson, D. C., "The Design and Application of Upwind Schemes on Unstructured Meshes," AIAA Paper 89–0366, January 1989.

[13] Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," *AIAA Journal*, Vol. 28, No. 8, August 1990, pp. 1381–1388.

[14] Frink, N. T., "A Fast Upwind Solver for the Euler Equations on Three-Dimensional Unstructured Meshes," AIAA Paper 91–0102, January 1991.

[15] Marcum, D. L., and Agarwal, R. K., "Finite Element Navier–Stokes Solver for Unstructured Grids," *AIAA Journal*, Vol. 30, No. 3, March 1992, pp. 648–654.

[16] Baldwin, B. S., and Barth, T. J., "A One–Equation Turbulence Transport Model for High Reynolds Number Wall–Bounded Flows," AIAA Paper 91–0610, January 1991.

[17] Spalart, P. R., and Allmaras, S. R., "A One–Equation Turbulence Model for Aerodynamic Flows," AIAA Paper 92–0439, January 1991.

[18] Rumsey, C. L., and Vatsa, V. N., "A Comparison of the Predictive Capabilities of Several Turbulence Models Using Upwind and Central–Difference Computer Codes," AIAA Paper 93–0192, January 1993.

[19] Sai, V. A., and Lutfy, F. M., "Analysis of the Baldwin–Barth and Spalart–Allmaras One–Equation Turbulence Models," *AIAA Journal*, Vol. 33, No. 10, October 1995, pp. 1971–1974.

[20] Lohner, R., Yang, C., Baum, J. D., "Rigid and Flexible Store Separation Simulations Using Dynamic Adaptive Unstructured Grid Technologies," *Proceedings of the First AFOSR Conference on Dynamic Motion CFD*, Rutgers University, New Brunswick, New Jersey, June 1996, pp. 1–29.

[21] Baysal, O., Singh, K. P., Yen, G. W., "Dynamic CFD Methods for Prescribed and Aerodynamically–Determined Relative–Moving Multibody Problems," *ibid*, pp. 33–44.

[22] Venkatakrishnan, V., and Mavriplis, D. J., "Computation of Unsteady Flows Over Complex Geometries in Relative Motion," *ibid*, pp. 93–112.

[23] Kleb, W. L., "Aerodynamic Characteristics of an Aerospace Vehicle During a Subsonic Pitch–Over Maneuver," AIAA Paper 96–0825, January 1996.

[24] Thomas, P. D., and Lombard, C. K., "Geometric Conservation Law and Its Application to Flow Computations on Moving Grids," *AIAA Journal*, Vol. 17, No. 10, October 1979, pp. 1030–1037.

[25] Warsi, Z. U. A., *Fluid Dynamics: Theoretical and Computational Approaches*, CRC Press, 1993.

[26] White, F. M., *Viscous Fluid Flow*, McGraw–Hill Inc., Second Edition, 1991.

[27] Hirt, C. W., Amsden, A. A., and Cook, J. L., "An Arbitrary Lagrangian–Eulerian Computing Method for All Flow Speeds," *Journal of Computational Physics*, Vol. 14, No. 3, March 1974, pp. 227–253 and reprinted in Vol. 135, No. 2, August 1997, pp. 203–216.

[28] Fortin, F., Jones, D. J., "Computations for Moving Stores Using Hybrid Grids," *Proceedings of the First AFOSR Conference on Dynamic Motion CFD*, Rutgers University, New Brunswick, New Jersey, June 1996, pp. 73–91.

[29] Farhat, C., Lesoinne, M., and Koobus, B., "A High Fidelity and High Performance Computational Methodology for the Solution of Transient Nonlinear Viscous Aeroelastic Problems," *ibid*, pp. 161–187.

[30] Alonso, J. J., Sheffer, S. G., Martinelli, L., and Jameson, A., "Parallel Unsteady Simulation of the Flow Through a Helicopter Rotor in Hover Including Aeroelastic Effects," *ibid*, pp. 189–201.

[31] Yang, H. Q., "A Strongly–Coupled Fluid Structure Interaction Method for Aeroelasticity," *ibid*, pp. 203–217.

[32] Slater, J. W., Chung, J., and Cole, G. L., "Computation of Unsteady Aeropropulsion Flows with Moving Geometry," *ibid*, pp. 269–279.

[33] Brenner, P., "Unsteady Flows About Bodies in Relative Motion," *ibid*, pp. 401–412.

[34] Bayyuk, S. A., Powell, K. G., van Leer, B., "An Algorithm for Simulation of Flows with Moving Boundaries and Fluid–Structure Interactions," *ibid*, pp. 413–424.

[35] Koomullil, R., Soni, B., and Huang, C., "Unsteady Flow Simulations on Hybrid Grids," *ibid*, pp. 425–434.

[36] Illinca, A., Camarero, R., Trepanier, J. Y., Reggio, M., "Error Estimator and Adaptive Moving Grids for Finite Volume Schemes," *AIAA Journal*, Vol. 33, No. 11, November 1995, pp. 2058–2065.

[37] Singh, K. P., Newman, J. C., and Baysal, O., "Dynamic Unstructured Method for Flows Past Multiple Objects in Relative Motion," *AIAA Journal*, Vol. 33, No. 4, April 1995, pp. 641–649.

[38] Margolin, L. G., "Introduction to (reprint of) "An Arbitrary Lagrangian–Eulerian Computing Method for All Flow Speeds," *Journal of Computational Physics*, Vol. 135, No. 2, August 1997, pp. 198–202.

[39] Anderson, W. K., and Bonhaus, D. L., "Aerodynamic Design on Unstructured Grids for Turbulent Flows," NASA Technical Memorandum 11287, June 1997.

[40] Beer, F. P., and Johnson, E. R. Jr., *Vector Mechanics for Engineers: Statics and Dynamics,* McGraw–Hill Book Company, Third Edition, 1977.

[41] Hibbeler, R. C., *Engineering Mechanics: Dynamics,* MacMillan Publishing Company, Inc., Fifth Edition, 1989.

[42] Stevens, B. L., and Lewis, F. L., *Aircraft Control and Simulation,* John Wiley & Sons, Inc., 1992.

[43] Bertin, J. J., and Smith, M. L., *Aerodynamics for Engineers,* Prentice–Hall, Inc., 1979.

[44] Anderson, J. D., Jr., *Fundamentals of Aerodynamics,* McGraw–Hill, Inc., Second Edition, 1991.

[45] Anderson, W. K., and Bonhaus, D. L., "An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids," *Computers Fluids,* Vol. 23, No. 1, 1994, pp. 1–21.

[46] Barth, T. J., "Numerical Aspects of Computing Viscous High Reynolds Number Flows on Unstructured Meshes," AIAA Paper 91–0721, January 1991.

[47] Anderson, J. D. Jr., *Computational Fluid Dynamics: The Basics with Applications,* McGraw–Hill, Inc., 1995.

[48] Van Leer, B., "Flux Vector Splitting for the Euler Equations," *Lecture Notes in Physics,* Vol. 170, Springer–Verlag, 1982.

[49] Anderson, W. K., Thomas, J. L., and Rumsey, C. L., "Extension of Applications of Flux–Vector Splitting to Unsteady Calculations on Dynamic Meshes," AIAA Paper 87–1152, June 1987.

[50] Anderson, W. K., "A Grid Generation and Flow Solver Solution Method for the Euler Equations on Unstructured Grids," *Journal of Computational Physics,* Vol. 110, 1994, pp. 23–38.

[51] Batina, J. T., "Implicit Flux–Split Euler Schemes for Unsteady Aerodynamic Analysis Involving Unstructured Dynamic Meshes," *AIAA Journal,* Vol. 29, No. 11, 1991, pp. 1836–1843.

[52] Parpia, I. H., "van Leer Flux Vector Splitting in Moving Coordinates," *AIAA Journal,* Vol. 26, No. 1, 1988, pp. 113–115.

[53] Tannehill, J. C., Anderson, D. A., and Pletcher, R. H., *Computational Fluid Mechanics and Heat Transfer,* Taylor & Francis, Second Edition, 1997.

[54] Hirsch, C., *Numerical Computation of Internal and External Flows, Volume 2: Computational Methods for Inviscid and Viscous Flows,* John Wiley & Sons, 1990.

APPENDIX A

INVISCID FLUX JACOBIANS

The flux Jacobians appear in the 3–D unstructured CFD method for dynamic motion as a result of the time linearizations and Newton's method. They are derived from taking the partial derivative of the inviscid flux vector with respect to either the conserved variable or primitive variable vector, depending on the formulation required. The net result is a five by five matrix, A, the flux Jacobian matrix.

The flux vector at the surface of a control volume is obtained by taking the dot product of the flux with the unit normal, that is

$$\vec{F} \cdot \hat{n} = \begin{bmatrix} \rho \overline{U} \\ \rho \overline{U} u + p n_x \\ \rho \overline{U} v + p n_y \\ \rho \overline{U} w + p n_z \\ (E + p)\overline{U} + p a_t \end{bmatrix} \tag{A.1}$$

where the contravariant velocity is

$$\overline{U} = \vec{V} \cdot \hat{n} = (u - x_t)n_x + (v - y_t)n_y + (w - z_t)n_z \tag{A.2}$$

and the contravariant face speed is

$$a_t = x_t n_x + y_t n_y + z_t n_z \tag{A.3}$$

The flux Jacobian is obtained by

$$\frac{\partial \vec{F}}{\partial \vec{Q}} = A \tag{A.4}$$

111